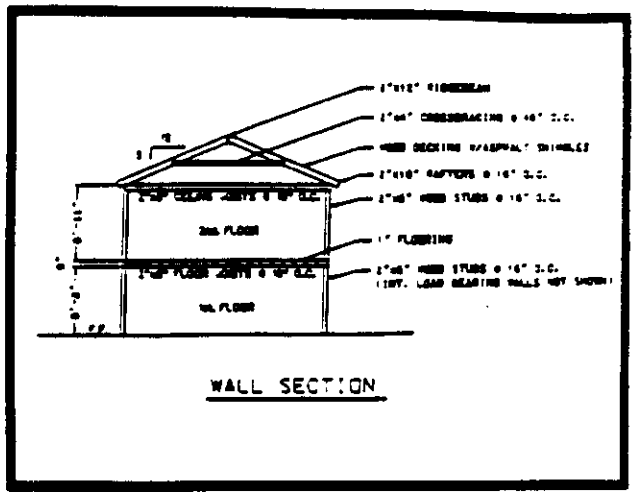
Protective Design - Mandatory Center of Expertise
Technical Report 92-4

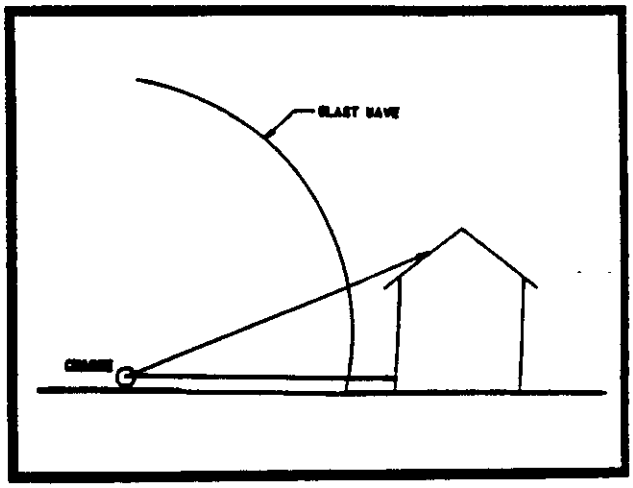
Facility And Component Explosive Damage Assessment Program

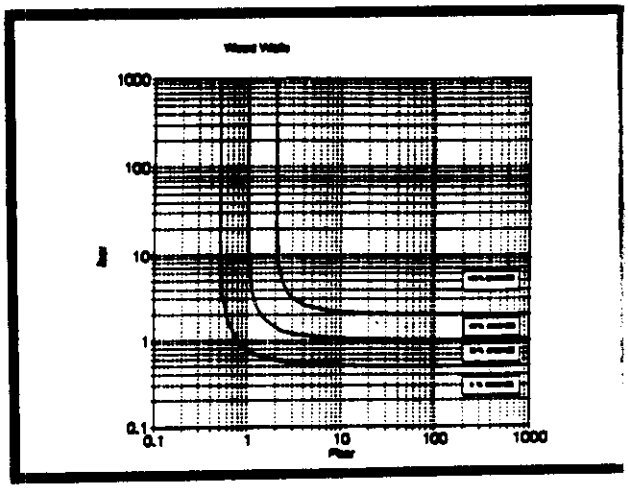
(FACEDAP)

Programmer's Manual

Version 1.2







SwRI Project No. 06-5145-001
Contract No. DACA 45-91-D-0019

Prepared for:

Department of the Army
Corps of Engineers, Omaha District
CEMRO-ED-ST
215 N. 17th Street
Omaha, Nebraska 68102-4978

Modified May 1994

Protective Design - Mandatory Center of Expertise
Technical Report 92-4

Facility And Component Explosive Damage Assessment Program

(FACEDAP)

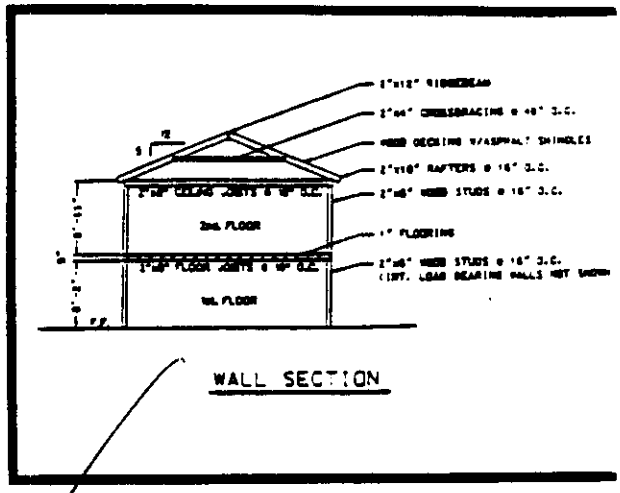
Programmer's Manual

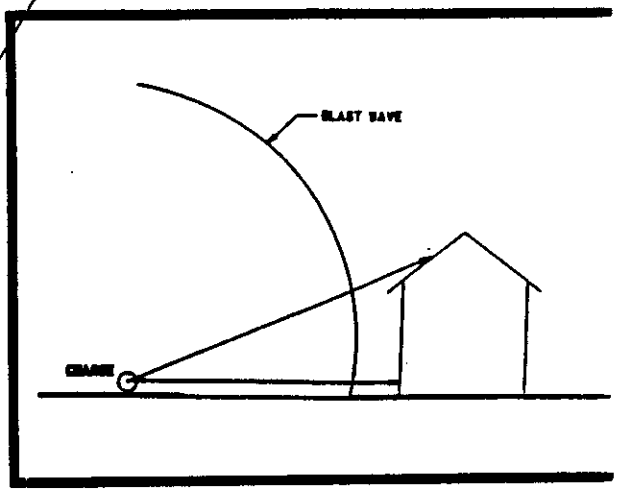
SwRI Project No. 06-5145-001
Contract No. DACA 45-91-D-0019

Prepared for:

Department of the Army
Corps of Engineers, Omaha District
CEMRO-ED-ST
215 N. 17th Street
Omaha, Nebraska 68102-4978

April 1993





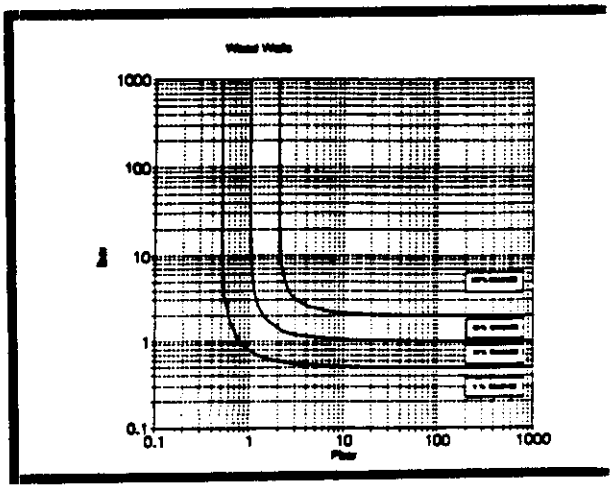


Table of Contents

Table of Contents
(Continued)

List of Figures

List of Tables

1.0 Introduction

The purpose of this manual is to familiarize the reader with the basic architecture of the FACEDAP computer code. The information in the manual includes: 1) a detailed flow diagram and description of the primary programs and subroutines in the FACEDAP code; 2) a description of the software used by the code; 3) a description of the matrices and arrays which have dimensions which control the maximum problem size that can be run in the FACEDAP code; and, 4) instructions on compiling and linking the primary executables in the code. This manual is intended to be read in conjunction with the FACEDAP User's Manual [1], the FACEDAP Theory Manual [2], the IOSUB User's Manual [3], and the ESHELL User's Manual [4]. The FACEDAP User's Manual [1] describes use of the FACEDAP program and includes; one worked example problem, thirteen example buildings with example input files that are included on the program disk, and a listing of all error messages with explanation. The FACEDAP Theory Manual [2] discusses and assesses the theoretical approach used to determine building vulnerability to blast damage in the FACEDAP program . The IOSUB User's Manual [3] and ESHELL User's Manual [4] describe subroutines in these software packages which are used to control screen display in the FACEDAP code and run the FACEDAP driver using a minimum amount of computer memory.

The FACEDAP program is an approximate procedure for determining the vulnerability of common structural components and industrial buildings to explosive threats. It is not intended for use in blast resistant design or in any other situations where high accuracy is required. If the single component analysis option is invoked, the program calculates the blast damage to an input component from an input explosive threat using a procedure which is based on available experimental data and basic dynamic structural response theory. If the building analysis option is invoked, the building vulnerability, in terms of the percentage of building damage caused by the explosive threat, is calculated by the program in a two-step procedure. In the first step, the damage to each component in the building is calculated. In the second step, the calculated damage to each component is summed and divided by a value corresponding to total building failure to determine the percentage of building damage. Building reusability and replacement factors and the level of protection provided to assets within the building are also calculated. The theoretical approach used by the FACEDAP program is assessed and discussed in detail in the FACEDAP Theory Manual[2]. As the FACEDAP User's Manual [1] explains, input of the relatively large amount of required building information is facilitated by the FACEDAP program preprocessor, which automatically generates many building components, only requires input of unique building component properties sets, and automatically generates "dependencies" that define how building components support, and are supported by, other components.

The FACEDAP computer code consists of a driver, FACEDAP.EXE, three primary processors; the Preprocessor, the Analysis program, the Postprocessor; and a minor executable and several subroutines which set default configuration values and retrieve existing input files. These processors are illustrated in Figure 1. The executables in each processor shown in Figure 1 are "child programs" which are run, one at a time, as designated by input from the user into the driver program. In general, each child program reads a file on the hard disk with required input information, performs various calculations using the input information and additional information input by the user, displays some calculated information on the screen, and writes an output file which is used as input by one of the other child programs. The FACEDAP driver also allows the user to retrieve

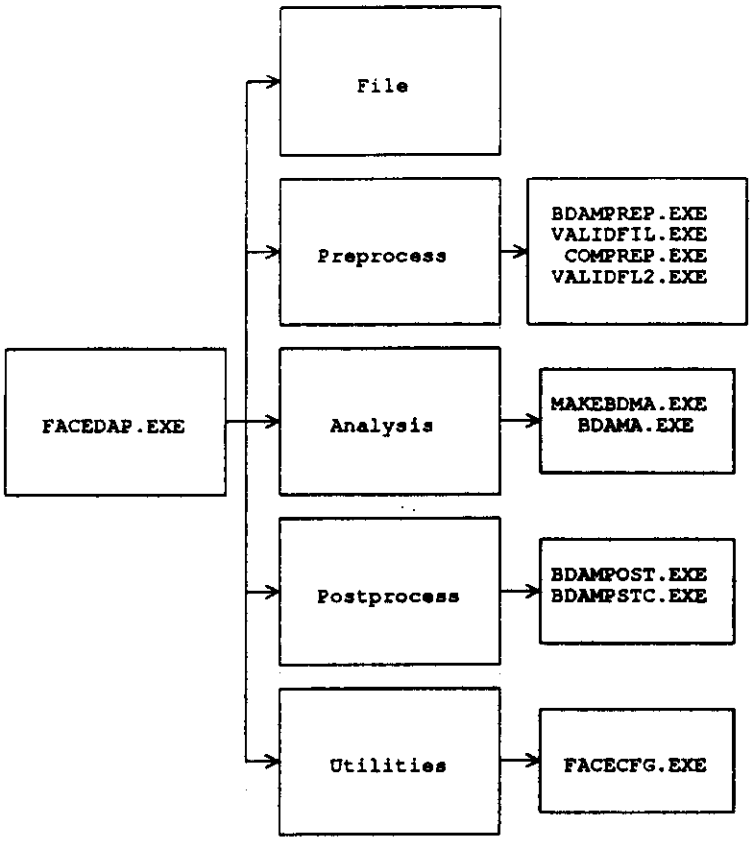


Figure 1. Flow Diagram for FACEDAP Program Driver

previous input files, establish program default directories, default printers, default print communications ports, set default screen colors, and shell out to DOS. These capabilities are included in the boxes marked "utilities" and "files" in Figure 1.

The FACEDAP Preprocessor reads all information defining the explosive charge weight and location and the components exposed to blast loading. In the case of a building analysis, the Preprocessor uses the fact that most buildings are comprised of a relatively small number of "unique" building components, which are used repetitively throughout the building construction, to reduce the effort required to input the required information. The user must initially break the building into large, planar wall/roof areas which are defined by four corner coordinates. All subsequent building input and output is initially defined in terms of one of the wall/roof areas and then in terms of components within that wall/roof area. This allows the user to more easily "manage" the relatively large amounts of detailed building information which must be input without the aid of a graphical display. It also allows the coordinates of building component endpoints to be input in terms of a local two-dimensional coordinate system in each wall/roof area rather than in terms of the global, three-dimensional coordinate system.

There are many checks built into the Preprocessor which issue warnings to the user during input if there appears to be an error in the input. The Preprocessor also makes extensive use of the IOSUB software, which has pre-coded subroutines that display user friendly form-type and spreadsheet-type screens to accept input. Finally, the Preprocessor includes a validation program which checks the user input and provides a specific error message for each error that is found. The FACEDAP program input can be simplified further in the future by the addition of a graphics package, which displays the spatial location and properties of input components. The availability of a graphics package would also enhance the program output.

If no errors are found by the validation program, the user can run the BDAMA.EXE program in the block labeled "Analysis" in Figure 1. This executable calculates building damage using the two-step procedure explained above. After the analysis has been completed, the Postprocessor displays building or single component blast damage information and blast load information. The Postprocessor makes extensive use of the IOSUB software to display blast damage information on each building component from a building analysis on spreadsheet-type screens in terms of the user-defined wall/roof areas of the building.

2.0 Flow Diagrams for the Primary Executables in the FACEDAP Code

Appendix A shows Calling Trees for each of the primary executables in the FACEDAP Code. Appendix A-1 shows the Calling Tree for the FACEDAP driver. Appendix A-2 shows the Calling Tree for BDAMPREP, the main Building Analysis Preprocessor executable. Appendix A-3 shows the Calling Tree for VALIDFIL, the primary Building Analysis error checking executable. Both BDAMPREP and VALIDFIL are part of the Preprocess module as shown in Figure 1. Appendix A-4 shows the Calling Tree for MAKEBDMA, which uses the information input into the Building and Single Component Preprocessors to write an input file for the BDAMA executable. Appendix A-5 shows the Calling Tree for BDAMA, which analyzes the building damage caused by the input explosive charge. Appendix A-6 shows the Calling Tree for the Building Analysis Postprocessor, BDAMPOST, which displays the calculated results. Appendices A-7 through A-9 show the Calling Trees for the Single Component Analysis executables. Appendix

A-7 is the Calling Tree for COMPPREP, which is the Single Component Analysis Preprocessor. The Single Component Analysis Validation routine, VALIDFL2, Calling Tree is shown in A-8. Appendix A-9 shows the Calling Tree for the Single Component Analysis Postprocessor, BDAMPSTC, which displays the component damage results.

The Calling Tree for the FACEDAP driver, in Appendix A-1, will be used as an example to illustrate how the Calling Trees in Appendix A and the subroutine purpose statements in Appendix B describe the flow of the FACEDAP program. The main program in the FACEDAP driver, which is also called FACEDAP, is the first program shown in the Calling Tree. This program calls the eleven indented subroutines shown below it. The order of these eleven subroutines is alphabetical, not the order in which the subroutines are invoked. All subroutines in the executables which call other subroutines follow the main program in the Calling Tree in alphabetical order. In the FACEDAP driver, two subroutines, namely CHGDIR1 and CHILD1, call other subroutines and therefore they are shown in alphabetical order under the main program with the (indented) subroutines they call. The calling arguments and the purpose of the subroutines shown in all the Calling Trees in Appendix A are listed and described in alphabetical order in Appendix B except as noted in the next two (2) paragraphs. The reader can understand the function CHGDIR1 by looking up this subroutine in Appendix B and reading the purpose statement. This standard method of presenting a program flow diagram is well suited for complex programs like the FACEDAP code which involve many subroutine calls.

Subroutines in the Calling Trees which are taken from the IOSUB software package without modification are described in the IOSUB User's Manual [3]. Only IOSUB subroutines which are used for overlay purposes, or are modified versions of the original IOSUB subroutine, are included in Appendix B. Filenames of modified IOSUB routines have a "B" proceeding the filename, such as BSPREAD.FOR instead of SPREAD.FOR. The subroutine call statement and the subroutine name within these files are not changed. The Calling Trees in Appendix A where generated by a useful IOSUB program called MKTRE. More information on this routine may be found in the IOSUB User's Manual.

Appendix C contains the routines which were added when the Single Component Analysis was added. Refer to Appendix C when an "*" appears following the subroutine/function name in the Calling Tree.

3.0 Software Used to Program the FACEDAP Code

The FACEDAP code is written in Microsoft Fortran, Version 5.1. As mentioned previously, many of the subroutines in the IOSUB software package are called by the FACEDAP code. These pre-coded subroutines, which are also written in Microsoft Fortran, control the most of the input and output in the code. Compiled versions of these subroutines are in the IOSUB libraries included on the FACEDAP Source Disk. A complete description of the IOSUB subroutines is included in Reference 3. Subroutines in the ESHELL software, which contain logic that minimizes the amount of memory used by the FACEDAP driver program, are linked with the driver program. This software enables all but approximately 3K bytes of the Driver program, FACEDAP.EXE, to be unloaded from memory while each child program is running, freeing more memory for the child programs.

None of the software in the ESHELL package is referenced in the Calling Tree in Appendix A-1. The ESHELL software, which is marketed by Kandu, Inc., is discussed in Reference 4. The IOSUB software is marketed by Business Systems Integration, in San Antonio, Texas.

4.0 Modifying the Problem Size Which Can be Analyzed with the FACEDAP Code

The FACEDAP Source Disk contains all program source files written or modified during this project, the ESHELL and IOSUB libraries which are linked with executables in the FACEDAP program, and batch files which contain the Microsoft Fortran compile and link commands necessary to build the program executables. Therefore, this disk contains all the code a user needs to modify the FACEDAP code and rebuild it. This section and the following sections contain programming information which helps explain the framework of the code. Most of the major matrices and arrays in the program are dimensioned using variables which are only defined once in an "include" file with the extension .INC. The only exceptions are the matrices in the BDAMA.EXE program, which are dimensioned in each subroutine where they are used as discussed at the end of this section. This makes it possible to change the size of arrays and matrices, and therefore the size of problem, by just changing the variables controlling the dimensions of the matrix in the appropriate include file. Table 1 shows the names of variables which define the dimensions of the major matrices and arrays, explains their use and their current values, and shows the files where they are located. Items 3, 5, and 6, in Table 1 refer to numbers in the SPRDHEAD.DAT and/or SPRDHED2.DAT data files which control the size of spreadsheets with input/output information that are displayed on the screen. These numbers are located in the header lines of the given spreadsheets in the SPRDHEAD.DAT and SPRDHED2 files. Spreadsheets are defined in this file by their numerical order in the file. **NOTE: SPRDHEAD.DAT is used by the Building Analysis executable. SPRDHED2.DAT is used by the Single Component Analysis executable.** The difference in these files is that the Single Component Analysis does not require Weighting Factors. Instead, a Span Length is required. In the Single Component Analysis the Span Length replaced the Weighting Factor unless the Span Length was already present, in which case the Weighting Factor was just deleted.

Table 2 shows the major matrices and arrays in the FACEDAP Code. It also shows the parameters in Table 1 which control the dimensions of the matrices, the variable type which is stored, and the purpose of each array or matrix. Major matrices in the BDAMA.EXE executable are shown separately in Table 3. The values of variables defining the maximum array and matrix dimensions in Table 2 have been carefully chosen so that most typical buildings can be input without exceeding the limits of the code. They have also been chosen to both limit the memory required by each separate child program so that the computer operating system and other typical background programs can run at the same time as the FACEDAP code, and allow each child program to maintain all required storage information in memory.

In spite of the efforts described above, it may still be necessary to increase the size of the maximum problem which can be analyzed with the FACEDAP code. The matrix dimensions which are considered most likely to require modification are those controlling the maximum number of components that can be input into the FACEDAP program in a single run and the sizes of the spreadsheet-type input/output screens. Item 1 in Table 1 shows that the FACEDAP code is currently set up so that it will analyze a maximum of 370 building components. This maximum causes the array, COMPG, which stores the building component information, to be very near the 64,000 byte limit for a matrix which is maintained in a single block of computer memory. As Tables 1 and 2

Table 1. Files Containing Dimensions Which Control Program Size

ITEM NO.	VARIABLE	FILE	VALUE	DESCRIPTION
1	max_num_components	BDAM.INC	370	Total components
2	max_wall_areas	BDAM.INC	50	Wall/Roof areas
3	Number of Rows in Input Screens for Building Geometry (Spreadsheet 26,28-34)	SPRDHEAD.DAT	150	Components per wall area
4	max_xmp_row	XMP.INC	150	Unique sets of component properties (total)
5	Number of Rows in Input Screens for Component Properties (Spreadsheets 1-24)	SPRDHEAD.DAT SPRDHED2.DAT	100	Component property sets per component type
6	Number of Rows in Spreadsheet 27 & max_generated_pairs	SPRDHEAD.DAT & BDAM.INC	50	Number of generated components from a master component
7	max_depend_pairs & dimension statements in BDAMA	BDAM.INC & BDAMA subroutines (see Table 3)	200	Total dependency pairs
8	max_column_ever	SPREAD.INC	17	Maximum columns on any spreadsheet
9	max_row_ever	SPREAD.INC	150	Maximum rows on any spreadsheet
10	total_spread_sheets	SPREAD.INC	34	Maximum number of spreadsheets
11	max_bdama_components & dimension statements in BDAMA	POSTPROCESSOR SUBROUTINES & BDAMA Subroutines (see Table 3)	400	Maximum number of components as allowed by BDAMA

Table 2. Major Arrays and Matrices in the FACEDAP Code

ITEM NO.	MATRIX NAME	VARIABLE TYPE	ROW DIMENSION	COLUMN DIMENSION	FILE WITH DIMENSIONING PARAMETERS	PURPOSE OF MATRIX
1	xmp	char*10	max_xmp_row	max_xmp_col	XMP.INC	Storage array for component properties
2	wall_table	char*11	max_row_ever	max_column_ever	WALLD.INC	Storage and spreadsheet array for wall/roof area definitions
3	compg	char*10	max_num_components	max_column_ever	COMPG.INC	Storage array for component geometry and dependency data
4	spread_table	int	17	total_spread_sheets	SPREAD.INC	Contains option instructions for each column of each spreadsheets
5	work_table	char*11	max_row_ever	max_column_ever	SPREADT.INC	Work array used for displaying spreadsheets
6	load_buf	char*10	4	N/A	LOADBUFF.INC	Form array used for load definitions
7	problem_buf	char*70	4	N/A	PROBBUFF.INC	Form array for title and descriptions
8	damage_buf	char*8	4	N/A	BDAMPOST.FOR	Form array for building damage
9	clink	int	max_xmp_row	N/A	CLINK.INC	Linklist for the XMP array
10	mlink	int	max_num_components	N/A	LINKLIST.INC	Linklist for the COMPG array
11	comp_def_buf	char*31	3	N/A	COMPDEF.INC	Form array used for component selection in the Single Component Analysis Program

show, COMPG is a 370 x 17 matrix, where each stored field is 10 characters, so that 62,900 bytes of memory is required to store this matrix. The COMPG matrix can be modified to store more building components by creating a separate matrix which stores the last four columns in the COMPG array and creating a pointer which links the corresponding rows in COMPG and in the new matrix. Column 1-13 in the COMPG matrix store component geometry data and columns 14-17 store the component dependencies.

Spreadsheet-type input screens, which are simply referred to as spreadsheets, are used in the Building and Single Component Preprocessors and the Building Postprocessor for input and output of building/component information. Each spreadsheet is defined by a Spreadsheet Number which refers to its position in the SPRDHEAD.DAT and SPRDHED2.DAT files. These files define the number of rows, number of columns, number of header lines, text in the column headers, text in the help messages, defaults input values and several other pieces of information for each spreadsheet. All spreadsheets, with the exception of WALL_TABLE, are displayed by the BSPREAD.FOR subroutine using the matrix, WORK_TABLE, which can be referred to as a work area. This work area is equivalenced to other matrices. However, the maximum dimension of any spreadsheet is governed by the dimensions of WORK_TABLE.

Items 3, 5, 6, in Table 1 show the current number of rows for each spreadsheet in the SPRDHEAD.DAT and/or SPRDHED2.DAT files. The maximum number of rows in any spreadsheet can be increased up to a maximum of 150 rows by changing the number of rows in the header line of the given spreadsheet in the SPRDHEAD.DAT and SPRDHED2 files. The limit for the maximum number of rows in a spreadsheet is set with the parameter max_row_ever in Table 1. The number of columns displayed to the screen for any given spreadsheet can be increased up to the limit of 17 columns using the steps shown below. The limit on the maximum number of columns in a spreadsheet is set with the parameter max_column_ever in Table 1.

1) Update SPRDHEAD.DAT and SPRDHED2.DAT - Change the number of columns in the header line of the given spreadsheet in the SPRDHEAD.DAT and SPRDHED2.DAT file. See the purpose statement for subroutine GETSPRED in Appendix B for an explanation of how the various spreadsheets and header lines are stored in these files.

2) Update OPTIONTB.DAT and OPTION2.DAT - These files contain information controlling special options that apply to each column in each spreadsheet. Each line in this file controls options for the spreadsheet in the SPRDHEAD.DAT and SPRDHED2.DAT files. The option control parameters on the second line, for example, control the second spreadsheet called by the program. Update the option control parameters for each new spreadsheet column as explained in the comment statements at the top of the BDAMPREP.FOR and BDAMPOST.FOR files. OPTIONTB.DAT is used for the Building Analysis and OPTION2.DAT is used for the Single Component Analysis.

3) Update GETVARS.FOR and GETVARS2.FOR - Column numbers used for default calculations in spreadsheets must be updated in GETVARS.FOR for the Building Analysis spreadsheets and GETVARS2.FOR for the Single Component Analysis spreadsheets.

4) Update CALCDEFS.FOR - Formulas used to calculate default component properties in the first 24 spreadsheets in SPRDHEAD.DAT and SPRDHED2.DAT must be updated with new variable numbers as per assignment in GETVARS.FOR and GETVARS2.DAT.

5) Update CONV4PRT.FOR - Update EXCLUDE_COL and MAX_COMP_COL arrays. EXCLUDE_COL is used to determine columns on the spreadsheet that are character types and therefore do not need conversion from character to real or integer. This routine is used to print the Preprocessor reports. NOTE: The data statement contains the excludes for the Building Analysis spreadsheets. These excludes are modified using assignment statements when this routine is used for a Single Component Analysis spreadsheets.

6) Update VALIDCMP - Update the EXCLUDE_COL and MAX_COMP_COL matrices. EXCLUDE_COL is used to determine which columns on the spreadsheet are excluded from the check for a 0 value. MAX_COMP_COL contains the maximum number of columns displayed on each spreadsheet and is used for looping control when checking columns with EXCLUDE_COL. NOTE: The data statement contains the excludes for the Building Analysis spreadsheets. These excludes are modified using assignment statements when this routine is used for a Single Component Analysis spreadsheets.

7) Re-link BDAMPREP using LXBDAMP.
Re-link COMPPREP using LXPREP2.

8) Re-link VALIDFIL using LXVALID.
Re-link VALIDFL2 using LXVALID2.

9) Update BDAMAWRI.FOR and subroutines in the BDAMA executable if the information read into the new columns affects these subroutines.

The ramifications of increasing the number of columns in a given spreadsheet above the current 17 column limit are much more extensive. All the storage matrices, XMP, COMPG, and WORK_TABLE, must be increased as well as several data files and subroutines. As previously pointed out, COMPG is already extremely close to the maximum size which can be contained in a single block of memory so it is not advisable to take this step with out making the suggested storage changes in COMPG.

The BDAMA.EXE executable, which is shown in the Analysis block in Figure 1, was written during a previous project and does not pass any information using common blocks. The sponsor of this work precluded the use of common blocks and therefore matrices and arrays are dimensioned separately in each subroutine where they are used. Modifications were made during the development of the FACEDAP code but they did not affect the structure of this executable. Table 3 shows the major matrices in this executable, the subroutines where they are dimensioned, and their current values.

Table 3. Matrices in BDAMA.EXE with Dimensions that Control Program Size

ITEM NO.	MATRIX	SUBROUTINE	DIMENSION
1	dam	COMPD.FOR COMPR.FOR OUTPOST.FOR SUM.FOR BDAM1.FOR	(400,8)
2	newdep	DEPMOD.FOR	(400,2)
3	idep	DEPMOD.FOR BDAM1.FOR DEPIN.FOR READAT.FOR	(200,2)

5.0 Compiling and Linking the FACEDAP Code

The subroutines shown in the Calling Trees in Appendix A control the input, output, and calculation scheme of the FACEDAP code. These files are all included on the FACEDAP Source Disk with root names which consist of the subroutine names shown in the Calling Trees followed by the extension .FOR. The Source Disk also includes files with the extensions .BAT, .LNK, and .INC, and .DAT, which are needed in order to compile and link the FACEDAP code. All the files required to build any of the FACEDAP code executables are in the archived file FACECODE.ZIP on the source disk. This disk includes a README.DOC file with instructions on how to unarchive the individual files from FACECODE.ZIP using the PKUNZIP.EXE file which is also on the source disk.

The .INC files contain common blocks that pass variables, arrays, and matrices between subroutines in the code (except in BDAMA.EXE where there are no common blocks). These files, which are called "include" files, are referenced with the fortran INCLUDE statement in all subroutines where the common blocks in the files are used. Information required by the FACEDAP code to set up input and output screens is programmed into files with the extension .DAT. The

.DAT files are ASCII character files which contain headings and help messages displayed by Preprocessor and Postprocessor screens as well as control parameters which define the size of each spreadsheet display.

The .BAT files contain the Microsoft Fortran compile and link commands necessary to build the executables in the FACEDAP code. During the development stage of the program, compilation was done using the FPC.BAT file provided on the Source Disk. This batch file contains the following two statements:

```
fl /FPi /Fs /Zi /Ge /Gt70 /c /4ccd /4Yb /Od /4Yd /G2 %1.for

rem fl /FPi /Fs /Zi /Ge /Gt70 /c /Od /4Yd /G2 %1.for
```

The first line contains the compilation statement for what is considered the full debugging mode. This statement causes extended error handling and bounds checking to be performed. The second line is a partial debug. It does not perform any bounds checking or extended error handling. Both, however, disable the compiler optimization and require that all variables be declared. Both methods generate the symbolic information required by the Microsoft CodeView Debugger. The remark command, REM, can be switched to remark out the command which is not used. The link method used while in the debugging mode also enables CodeView. The following statement shows the link text used.

```
link /CO /SE:512
```

The size of the executable generated in these modes is much larger than that generated in the production mode. The compilation command for the production mode, which is shown below, is contained in FPCPRO.BAT on the Source Disk.

```
fl /FPi /Gt70 /G2 /c %1.for
```

This command invokes the compiler optimizer and removes all Code View requirements. The link used for the production mode packs the executable in order to reduce the size of the executable.

```
link /SE:512 /EXEPACK
```

Table 4 shows the link batch files (LX*.BAT files) required to create a new version of each of the executables included in the FACEDAP code. This table also shows the files created and read by each executable during execution. Most of the files which are created during execution have a user designated root name (the name entered as the "save file" name) which is referred to Table 4 with an asterisk. The LX*.BAT files have both debug mode and production mode link statements in them. One of these statement should always be remarked out.

Most of these link batch files invoke a *.LNK file, which is the link response file. The asterisk refers to the name of the executable which is generated with the file. These link response files list all the subroutines and the IOSUB and ESHELL libraries that are required in the link command which creates the executable. Prior to running any of the link batch files, the user must compile all .FOR files required by the executable using Microsoft Fortran. The .INC files referenced in the subroutines that are linked, as well as IOSUB and ESHELL libraries on the FACEDAP Source Disk, must also be available in the directory where the executable is linked .

To facilitate having the required files for generating an executable, batch files were created to extract the required files from the FACECODE.ZIP source archive. By running the appropriate batch file, all FORTRAN, include, libraries and batch files necessary for the executable generation, will be extracted and placed in the current directory. This prevents having to have all source files present when only one executable is to be modified.

Table 4. Information on Linking Major FACEDAP Executables and Output Files Created During Program Execution

EXECUTABLE FILE NAME	EXTRACT BATCH FILE	LINK BATCH FILE	INPUT FILES	OUTPUT FILES
FACEDAP.EXE	XFACE.BAT	LXFACE.BAT	FACE.CFG MAINTITL.DAT *.BLG	N/A
BDAMPREP.EXE	XPREPB.BAT	LXBDAMP.BAT	FACE.CFG OPTIONTB.DAT SPRDHEAD.DAT PRBTITLE.FRM LOADDEF.FRM *.BLG	*.BLG *.REP
VALIDFIL.EXE	XVALIDB.BAT	LXVALID.BAT	FACE.CFG *.BLG	*.BLG *.ERR
MAKEBDMA.EXE	XMAKEB.BAT	LXMAKEB.BAT	FACE.CFG *.BLG	BDAMA.IN
BDAMA.EXE	XBDAMA.BAT	LXBDAMA.BAT	*.BLG BDAMA.IN	*.PST
BDAMPOST.EXE	XPOSTB.BAT	LXPOST2.BAT	FACE.CFG OPTIONTB.DAT SPRDHEAD.DAT BUILDAM2.FRM LOADPOST.FRM *.BLG *.PST	*.REP
COMPPREP.EXE	XPREPC.BAT	LXPREP2.BAT	FACE.CFG OPTION2.DAT SPRDHED2.DAT PRBTITLE.FRM LOADCOMP.FRM COMPDEF.FRM COMP*.BLG	COMP*.BLG COMP*.REP
VALIDFL2.EXE	XVALIDC.BAT	LXVALID2.BAT	FACE.CFG COMP*.BLG	COMP*.BLG COMP*.ERR
BDAMPSTC.EXE	XPOSTC.BAT	LXPOSTC.BAT	FACE.CFG BUILDAMC.FRM LOADCMPP.FRM COMP*.BLG COMP*.PST	COMP*.REP

* Indicates User assigned problem name, which is assigned in the Preprocessor, BDAMPREP.EXE or COMPPREP.EXE.

References

1. Oswald, C.J., and Skerhut, D., "FACEDAP User's manual," Contract No. DACA 45-91-D-0019, U.S. Army Corps of Engineers, Omaha District, April 1993.

2. Oswald, C.J., "FACEDAP Theory Manual," Contract No. DACA 45-91-D-0019, U.S. Army Corps of Engineers, Omaha District, April 1993.

3. IOSUB: A Library of Input and Output SUBroutines for use with Microsoft® Fortran, Version 2.0 User's Manual, Business Systems Integration, San Antonio, TX.

4. Empty Shell Version 1.1 Documentation, Kandu, Inc., Hamilton, NJ.

Appendix A

Indented Calling Trees for the FACEDAP Program

Appendix A-1

Indented Calling Tree for the FACEDAP Driver

```
facedap
      chgdir1
      child
      clng
      cls
      curof
      curon
      dialog
      erabox
      fclose
      fdel
      fopen
      fsize
      getlun
      getmod
      hlpmsg
      idtmc
      imcoff
      inpfil
      isxlim
      isylim
      msgbox
      popmen
      putstr
      rdcfg
      stdmsg
      wrcfg
chgdir1
      chdir
      clng
      dialog
      edtfld
      erabox
      getdir
      mkdir
      msgbox
child1
      clng
      cls
      espath
      getcur
      getmod
      hlpmsg
      idtmc
      imcoff
```

Appendix A-2

Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module

```
bdamprep
     buildgeo
     charge
     chgatr
     chgdirl
     clng
     cls
     compprop
     curof
     curon
     dialog
     editform
     erabox
     filexist
     getarg
     getdata
     getlun
     getmod
     hlpmsg
     i2chr
     idtmc
     imcoff
     inpfil
     isxlim
     isylim
     msgbox
     prepprnt
     putstr
     rdcfg
     repasc
     reset
     savedata
     setmod
     showme
     ucase
bselect
     chrwin
     clng
     drawbo
     fucase
     getmod
     ismpos
     putstr
     readke
     repasc
     sort_st
     srchf
     stdmsg
     winchr
bsetmsk
     repasc
bshowme
     chrwin
     cmpos
     drawbo
     fucase
     getmod
     putstr
     readke
     stdmsg
     ucase
     winchr
bspread
     chgatr
     chrwin
     clng
     curof
     curon
     dialog
     drawbo
     drawhl
     drawvl
     erabox
     getinp
     getmod
     hlpmsg
     idtmc
     imcoff
     msgbox
     putstr
     repasc
     rightj
     spreadop
     strwin
     winchr
     winstr
buildgeo
     cdepend
     clng
     cls
     compdef
     dialog
     erabox
     framdef
     getwname
     hlpmsg
     i2chr
     leftj
     msgbox
     putstr
     showme
     strwin
     walldef
     winstr
     wnindex
     xyz2
     xyzcoord
calcdefs
     clng
     cperror
     hlpmsg
     rightj
cdepend
     clng
     cls
     erabox
     gendpnd
     getdpnd
     getspred
     hlpmsg
     i2chr
     ktdpnd
     ktwdpnd
     msgbox
     ptdepend
     putstr
     rightj
     spread
     strwin
     winstr
centerln
     clng
     hlpmsg
     leftj
     repasc
charge
     clng
     editform
     erabox
     hlpmsg
chgdirl
     chdir
     clng
     dialog
     edtfld
     erabox
     getdir
     mkdir
     msgbox
chk2way
     clng
     gtxmprow
     hlpmsg
     leftj
     wnindex
     xyz2
chkcomp
     clng
     hlpmsg
     i2chr
     itypechk
     leftj
     rightj
chkmastr
     clng
     delgen
     dialog
     gtcompgr
     leftj
compdef
     chkcomp
     chkmastr
     clng
     cls
     dialog
     gen
     genit
     getspred
     gtgroup
     hlpmsg
     leftj
     msgbox
     ptgroup
     putstr
     repasc
     spread
     stdmsg
     strwin
     updatgen
     winstr
     xylimit
```

Appendix A-2

Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module (continued)

```
compprop
     clng
     cls
     erabox
     getspred
     gtmat
     hlpmsg
     i2chr
     leftj
     msgbox
     ptmat
     putstr
     showme
     spread
     strwin
     ucase
     winstr
conv4prt
     clng
     hlpmsg
cperror
     hlpmsg
cprphead
     repasc
cprpprnt
     clng
     cprprep
     erabox
     hlpmsg
     msgbox
     showme
     stdmsg
     strwin
     winstr
cprprep
     centerln
     clng
     conv4prt
     cprphead
     csort2d
     hlpmsg
     i2chr
     leftj
     repasc
csort2d
delgen
     clng
     dellink
     leftj
dellink
delwall
     clng
     fndwcomp
     gtgroup
editform
     optchk
     optionb
     readf
filexist
     clng
     erabox
     getinp
     hlpmsg
     msgbox
     putstr
     showme
     ucase
filldp
     clng
     i2chr
     rightj
fillgeo
     clng
     csort2d
     fndgen
     fndwcomp
     hlpmsg
     leftj
     rightj
fndcomp
     clng
     cls
     erabox
     hlpmsg
     i2chr
     leftj
     msgbox
     select
fndgen
     clng
     leftj
fndwcomp
     clng
     leftj
framdef
     chkcomp
     clng
     cls
     getspred
     gtgroup
     hlpmsg
     leftj
     msgbox
     ptgroup
     putstr
     readke
     spread
     stdmsg
     wnindex
     xylimit
gen
     chr2i
     clng
     hlpmsg
     i2chr
     leftj
     r2chr
     setcid
gendpnd
     clng
     gtcompgr
     gtxmprow
     hlpmsg
     iroof
     leftj
     wnindex
     xyz2
genit
     clng
     cls
     getspred
     gtgroup
     ptgroup
     putstr
     spread
geomprnt
     centerln
     clng
     fillgeo
     leftj
     prephead
     repasc
getdata
     getlun
     hlpmsg
getdpnd
     clng
     filldp
     fndgen
     fndwcomp
     hlpmsg
     leftj
getinp
     addper
     chr2i
     clng
     edtfld
     fucase
     funits
     getmod
     putstr
     readke
     repasc
     setcur
     setmsk
     stdmsg
     strwin
     tone
     valida
     winstr
getspred
     clng
     getlun
     hlpmsg
     repasc
getvars
getwalls
     clng
     hlpmsg
     iroof
     msgbox
     select
getwname
     clng
     erabox
     hlpmsg
     leftj
     msgbox
     select
gtcompgr
     clng
     leftj
```

Appendix A-2

Indented Calling Tree for BDAMPREP.EXE in the FACEDAP Preprocess Module (continued)

```
gtfulnam
      hlpmsg
      leftj
gtgroup
      clng
      dellink
      fndwcomp
      hlpmsg
      leftj
      rightj
gtmat
      dellink
      i2chr
gtxmprow
      clng
      hlpmsg
      leftj
inslink
      hlpmsg
iroof
      clng
      hlpmsg
itypechk
      chk2way
      clng
      hlpmsg
      iroof
      wnindex
ktdpnd
      clng
ktwdpnd
      clng
      leftj
leftj
      clng
      hlpmsg
      repasc
optchk
optionb
      clng
      erabox
      showme
prephead
      repasc
prepprnt
      clng
      cprpprnt
      dpndprnt
      erabox
      geomprnt
      getlun
      getwname
      hlpmsg
      leftj
      msgbox
      probrep
      showme
      stdmsg
      strwin
      wallrep
      winstr
ptdepend
      clng
      gtcompgr
      hlpmsg
      leftj
ptgroup
      chr2i
      clng
      hlpmsg
      inslink
      leftj
      setcid
      setfid
ptmat
      clng
      i2chr
      inslink
readf
      addper
      clng
      getinp
      getmod
      ismpos
      mkfrm
      msgbox
      putstr
      readke
      repasc
      select
      setmsk
      strwin
      swpcol
      winstr
reset
      repasc
rightj
      clng
      hlpmsg
      repasc
rsort2d
samecord
savedata
      clng
      getlun
setcid
      clng
      i2chr
setfid
      i2chr
spreadop
      calcdefs
      clng
      drawbo
      fndcomp
      getvars
      getwalls
      gtcompgr
      gtfulnam
      hlpmsg
      leftj
      putstr
      repasc
      rightj
      showme
      stdmsg
      strwin
      winstr
updatgen
      clng
      fndwcomp
      gen
      gtgroup
      ptgroup
wallchk
      clng
      hlpmsg
      leftj
walldef
      clng
      cls
      delwall
      getlun
      hlpmsg
      i2chr
      leftj
      msgbox
      putstr
      repasc
      samecord
      spread
      ucase
      wallchk
      wallok
wallok
      clng
      hlpmsg
      i2chr
      iroof
      leftj
wallrep
      centerln
      clng
      csort2d
      hlpmsg
      leftj
      prephead
      repasc
wnindex
      clng
      gtcompgr
      hlpmsg
      leftj
xylimit
      clng
      hlpmsg
xyz2
      hlpmsg
xyzcoord
      clng
      hlpmsg
      leftj
      wnindex
```

Appendix A-3

Indented Calling Tree for VALIDFIL.EXE in the FACEDAP Preprocess Module

```
validfil
     chgatr
     clng
     cls
     curof
     curon
     getarg
     getdata
     getlun
     getmod
     hlpmsg
     i2chr
     idtmc
     imcoff
     isxlim
     isylim
     ktdpnd
     msgbox
     putstr
     rdcfg
     repasc
     review
     savedata
     setmod
     ucase
     validchg
     validcmp
     validpnd
     validwal
     validwcmp
bldgchk
     clng
     hlpmsg
     iroof
chgloc
chk2way
     clng
     gtxmprow
     hlpmsg
     leftj
     wnindex
     xyz2
chkcomp
     clng
     hlpmsg
     i2chr
     itypechk
     leftj
     rightj
compok
     clng
     fndgen
     hlpmsg
     leftj
fndgen
     clng
     leftj
getdata
     getlun
     hlpmsg
gtcompgr
     clng
     leftj
gtxmprow
     clng
     leftj
iroof
     clng
     hlpmsg
itypechk
     chk2way
     clng
     hlpmsg
     iroof
     wnindex
ktdpnd
     clng
leftj
     clng
     hlpmsg
     repasc
onorm
rightj
     clng
     hlpmsg
     repasc
samecord
savedata
     clng
     getlun
validchg
     chgloc
     clng
     hlpmsg
     iroof
     onorm
validcmp
     clng
     i2chr
     leftj
validpnd
     clng
     gtcompgr
     i2chr
     ktdpnd
     leftj
validwal
     bldgchk
     clng
     leftj
     wallok
validwcm
     chkcomp
     clng
     compok
     leftj
     samecord
     wnindex
     xylimit
     xyz2
     xyzcoord
wallok
     clng
     hlpmsg
     i2chr
     iroof
     leftj
wnindex
     clng
     gtcompgr
     hlpmsg
     leftj
xylimit
     clng
     hlpmsg
xyz2
     hlpmsg
xyzcoord
     clng
     hlpmsg
     leftj
     wnindex
```

Appendix A-4

Indented Calling Tree for MAKEBDMA.EXE in the FACEDAP Analysis Module

```
makebdma
     bdamawri
     bldgchk
     clng
     cls
     curof
     curon
     getarg
     getdata
     getlun
     getmod
     hlpmsg
     idtmc
     imcoff
     isxlim
     isylim
     rdcfg
     setmod
     ucase
bdamawri
     clng
     getlun
     gtxmprow
     hlpmsg
     ktdpnd
     leftj
     rsort2d
     wnindex
     xyz2
     xyzcoord
bldgchk
     clng
     hlpmsg
     iroof
getdata
     getlun
     hlpmsg
gtcompgr
     clng
     leftj
gtxmprow
     clng
     leftj
iroof
     clng
     hlpmsg
ktdpnd
     clng
leftj
     clng
     hlpmsg
     repasc
makebdma
     bdamawri
     bldgchk
     clng
     cls
     curof
     curon
     getarg
     getdata
     getlun
     getmod
     hlpmsg
     idtmc
     imcoff
     isxlim
     isylim
     rdcfg
     setmod
     ucase
rightj
     clng
     hlpmsg
     repasc
rsort2d
wnindex
     clng
     gtcompgr
     hlpmsg
     leftj
xyz2
     hlpmsg
xyzcoord
     clng
     hlpmsg
     leftj
     wnindex
```

Appendix A-5

Indented Calling Tree for BDAMA.EXE in the FACEDAP Analysis Module

```
bdama
     bdam1
bdam1
     clng
     compd
     compr
     depin
     depmod
     getarg
     outpost
     readat
     sum
blastpi
     reflect
     sideon
center
chgloc
compd
     blastpi
     center
     chgloc
     interp
     onorm
compr
depin
depmod
intcol
interp
intjst
     intcol
intstl
intwd
mapili
mar1wi
mar2wi
mau1wi
mau2wi
onorm
outpost
rc1wi
rc2wi
rcbmi
rceci
rcici
rcmrfi
rcpsi
readat
     mapili
     mar1wi
     mar2wi
     mau1wi
     mau2wi
     rc1wi
     rc2wi
     rcbmi
     rceci
     rcici
     rcmrfi
     rcpsi
     stbmi
     stcdi
     steci
     stici
     stmrfi
     stmswi
     stowji
     wdbmi
     wdeci
     wdici
     wdrfi
     wdwli
reflect
sideon
stbmi
stcdi
steci
stici
stmrfi
stmswi
stowji
sum
threat
wdbmi
wdeci
wdici
wdrfi
wdwli
```

Appendix A-6

Indented Calling Tree for BDAMPOST.EXE in the FACEDAP Postprocess Module

```
bdampost
    bload
    chgatr
    clng
    cls
    curof
    curon
    damagec
    dialog
    drawbo
    editfrm2
    erabox
    genframd
    getarg
    getdata
    getlun
    getmod
    hlpmsg
    i2chr
    idtmc
    imcoff
    isxlim
    isylim
    leftj
    msgbox
    postprnt
    putstr
    rdcfg
    repasc
    setmod
    showme
    stdmsg
    ucase
bload
    clng
    cls
    editfrm2
    erabox
    fillblst
    getspred
    getwname
    hlpmsg
    leftj
    msgbox
    putstr
    showme
    spread
    strwin
    winstr
blstrep
    centerln
    clng
    fillblst
    leftj
    repasc
    wrthead
bselect
    chrwin
    clng
    drawbo
    fucase
    getmod
    ismpos
    putstr
    readke
    repasc
    sort_st
    srchf
    stdmsg
    winchr
bsetmsk
    repasc
bshowme
    chrwin
    cmpos
    drawbo
    fucase
    getmod
    putstr
    readke
    stdmsg
    ucase
    winchr
bspread
    chgatr
    chrwin
    clng
    curof
    curon
    dialog
    drawbo
    drawhl
    drawvl
    erabox
    getinp
    getmod
    hlpmsg
    idtmc
    imcoff
    msgbox
    putstr
    repasc
    rightj
    spreadop
    strwin
    winchr
    winstr
buildrep
    centerln
    clng
centerln
    clng
    hlpmsg
    leftj
    repasc
csort2d
damagec
    clng
    cls
    erabox
    fillcdam
    fillmdam
    getspred
    getwname
    hlpmsg
    leftj
    msgbox
    putstr
    showme
    spread
    strwin
    winstr
editfrm2
    optchk
    optionb2
    readf
fillblst
    clng
    csort2d
    hlpmsg
    leftj
    rightj
fillcdam
    clng
    csort2d
    hlpmsg
    leftj
    rightj
fillmdam
    csort2d
    hlpmsg
    leftj
    rightj
framekil
    clng
    gtxmprow
    hlpmsg
    leftj
    wnindex
    xyz2
    xyzcoord
genframd
    clng
    framekil
    gtcompgr
    gtsortid
    hlpmsg
    leftj
    msgbox
getdata
    getlun
    hlpmsg
getspred
    clng
    getlun
    hlpmsg
    repasc
getwname
    clng
    erabox
    hlpmsg
    leftj
    msgbox
    select
gtcompgr
    clng
    leftj
gtfulnam
    hlpmsg
    leftj
gtsortid
```

Appendix A-6

Indented Calling Tree for BDAMPOST.EXE in the FACEDAP Postprocess Module (continued)

```
      hlpmsg
gtxmprow
      clng
      leftj
leftj
      clng
      hlpmsg
      repasc
mdamrep
      centerln
      clng
      fillmdam
      leftj
      wrthead
optchk
optionb2
postprnt
      blstrep
      buildrep
      erabox
      getwname
      hlpmsg
      leftj
      mdamrep
      msgbox
      showme
      stdmsg
      strwin
      wdamrep
      winstr
rightj
      clng
      hlpmsg
      repasc
spredop2
      clng
      drawbo
      gtfulnam
      hlpmsg
      putstr
      repasc
      stdmsg
      strwin
      winstr
wdamrep
      centerln
      clng
      fillcdam
      leftj
      repasc
      wrthead
wnindex
      clng
      gtcompgr
      hlpmsg
      leftj
wrthead
      repasc
xyz2
      hlpmsg
xyzcoord
      clng
      hlpmsg
      leftj
      wnindex
```

Appendix A-7

Indented Calling Tree for COMPREP.EXE in the FACEDAP Preprocess Module

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "".*

```
bsetmsk
     repasc
bshowme
     chrwin
     cmpos
     drawbo
     fucase
     getmod
     putstr
     readke
     stdmsg
     ucase
     winchr
bspread
     chgatr
     chrwin
     clng
     curof
     curon
     dialog
     drawbo
     drawhl
     drawvl
     erabox
     getinp
     getmod
     hlpmsg
     idtmc
     imcoff
     msgbox
     putstr
     repasc
     rightj
     spreadop
     strwin
     winchr
     winstr
calcdefs
     clng
     cperror
     hlpmsg
     rightj
centerln
     clng
     hlpmsg
     leftj
     repasc
chgdirl
     chdir
     clng
     dialog
     edtfld
     erabox
     getdir
     mkdir
     msgbox
compprep*
     chgatr
     chgdirl
     clng
     cls
     compprnt
     compprop
     curof
     curon
     dialog
     editform
     erabox
     filxist2*
     getarg
     getcomp*
     getlun
     getmod
     hlpmsg
     i2chr
     idtmc
     imcoff
     initcomp*
     inpfil2*
     isxlim
     isylim
     leftj
     msgbox
     putstr
     rdcfg
     repasc
     resetc*
     savecomp*
     setmod
     showme
     ucase
compprnt
     clng
     cprpprnt
     cselrep
     erabox
     getlun
     hlpmsg
     msgbox
     probrep2*
     showme
     stdmsg
     strwin
     winstr
compprop
     clng
     cls
     erabox
     getspred
     gtmat
     hlpmsg
     i2chr
     leftj
     msgbox
     ptmat
     putstr
     showme
     spread
     strwin
     ucase
     winstr
conv4prt
     clng
     hlpmsg
cperror
     hlpmsg
cprphed2*
     repasc
cprpprnt
     clng
     cprprep
     erabox
     hlpmsg
     msgbox
     showme
     stdmsg
     strwin
     winstr
cprprep
     centerln
     clng
     conv4prt
     cprphead
     csort2d
     hlpmsg
     i2chr
     leftj
     repasc
cselrep
     centerln
     clng
     leftj
csort2d
dellink
editform
     optchk
     optionb
     readf
filxist2*
     clng
     erabox
     getinp
     hlpmsg
     msgbox
     putstr
     showme
     ucase
fndcomp
     clng
     cls
     erabox
     hlpmsg
     i2chr
     leftj
     msgbox
     select
getcomp*
     getlun
     hlpmsg
getdefct*
     chr2i
     hlpmsg
getdefmt*
     chr2i
getinp
     addper
     chr2i
     clng
     edtfld
     fucase
```

Appendix A-7

Indented Calling Tree for COMPREP.EXE in the FACEDAP Preprocess Module (Continued)

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "".*

```
        funits
        getmod
        putstr
        readke
        repasc
        setcur
        setmsk
        stdmsg
        strwin
        tone
        valida
        winstr
getsprd2*
        clng
        getlun
        hlpmsg
        repasc
getvars2*
getwalls
        clng
        hlpmsg
        iroof
        msgbox
        select
gtcompgr
        clng
        leftj
gtfulnam
        hlpmsg
        leftj
gtmat
        dellink
        i2chr
gtshort*
        hlpmsg
        leftj
gtxmprow
        clng
        hlpmsg
        leftj
initcomp*
        chr2i
inpfil2*
        clng
        fcount
        fdir
        hlpmsg
        select
inslink
        hlpmsg
iroof
        clng
        hlpmsg
leftj
        clng
        hlpmsg
        repasc
optchk
optionc*
        clng
        cls
        erabox
        fndcomp
        getdefct
        getdefmt
        gtshort*
        hlpmsg
        putstr
        showme
        strwin
        winstr
probrep2*
        centerln
        clng
ptmat
        clng
        i2chr
        inslink
readf
        addper
        clng
        getinp
        getmod
        ismpos
        mkfrm
        msgbox
        putstr
        readke
        repasc
        select
        setmsk
        strwin
        swpcol
        winstr
resetc
        repasc
rightj
        clng
        hlpmsg
        repasc
savecomp*
        chr2i
        clng
        getlun
        hlpmsg
        leftj
spreadop
        calcdefs
        clng
        drawbo
        fndcomp
        getvars
        getwalls
        gtcompgr
        gtfulnam
        hlpmsg
        leftj
        putstr
        repasc
        rightj
        showme
        stdmsg
        strwin
        winstr
```

Appendix A-8

Indented Calling Tree for VALIDFL2.EXE in the FACEDAP Preprocess Module

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "".*

```
getcomp*
      getlun
      hlpmsg
gtshort*
      hlpmsg
      leftj
gtxmprow
      clng
      hlpmsg
      leftj
leftj
      clng
      hlpmsg
      repasc
savecomp*
      chr2i
      clng
      getlun
      hlpmsg
      leftj
validcmp
      clng
      i2chr
      leftj
validfl2*
      chgatr
      chr2i
      clng
      cls
      curof
      curon
      getarg
      getcomp*
      getlun
      getmod
      gtshort*
      gtxmprow
      hlpmsg
      i2chr
      idtmc
      imcoff
      isxlim
      isylim
      leftj
      msgbox
      putstr
      rdcfg
      repasc
      review
      savecomp*
      setmod
      ucase
      validcmp
```

Appendix A-9

Indented Calling Tree for BDAMPSTC.EXE in the FACEDAP Postprocess Module

Refer to Appendix C for descriptions of subroutines in Calling Tree followed by "".*

```
bdampstc*
      bloadc*
      chgatr
      clng
      cls
      comprep*
      curof
      curon
      dialog
      drawbo
      editfrm2
      erabox
      getarg
      getcomp*
      getlun
      getmod
      hlpmsg
      i2chr
      idtmc
      imcoff
      isxlim
      isylim
      leftj
      msgbox
      putstr
      rdcfg
      repasc
      setmod
      showme
      stdmsg
      ucase
bloadc*
      editfrm2
      erabox
      hlpmsg
      msgbox
      showme
      strwin
      winstr
bselect
      chrwin
      clng
      drawbo
      fucase
      getmod
      ismpos
      putstr
      readke
      repasc
      sort_st
      srchf
      stdmsg
      winchr
bsetmsk
      repasc
bshowme
      chrwin
      cmpos
      drawbo
      fucase
      getmod
      putstr
      readke
      stdmsg
      ucase
      winchr
centerln
      clng
      hlpmsg
      leftj
      repasc
compret*
      centerln
      clng
editfrm2
      optchk
      optionb2
      readf
getcomp*
      getlun
      hlpmsg
leftj
      clng
      hlpmsg
      repasc
optchk
optionb2
rightj
      clng
      hlpmsg
      repasc
```

Appendix B

Description of Primary Subroutines Called in the FACEDAP Code for Building Analysis and Single Component Analysis

***NOTE*: Subroutines Unique to Single Component Analysis are Shown in Appendix C**

```
------------------------------------------------------------------------
call  bdam1

 last modified --
  11/08/91  (tkb)
  11/19/92  (dds) - added call to outpost; opened file bdama.pst

 purpose --
   sets up building analysis through component damage summation.

 method --
   calls various menu subroutines to obtain desired information
   about problem to be solved.

 input --
   input data file
   and tape15 (direct access file with individual element data)

 output --
   tape15:  same file as input

 restrictions
   none set in bdam1
------------------------------------------------------------------------
call  bdamawri ()
function:    writes output from preprocessor in output file in same
             format used for bdama.exe read statements
             this file also creates this sortcompg matrix which must
             be stored on disk because it links compg in its final
             arrangement with the id numbers used by bdama
             this info will be needed by post-processor
parameters:
------------------------------------------------------------------------
call  blastpi( r, w, angle, pres, imp)

 last modified --
   6/11/91   (jpp)

 purpose --
   determination of blast pressure and impulse load on a component

 method --
   uses fitted functions from arbrl-tr-02555 (used to fit air blast
   curves in arlcd-sp-84001) to calculate blast pressure and impulse

 input --
   no files, just variables in argument list

 output --
   no files, just variables in argument list

 restrictions --
   r/w**(1/3) must lie between certain limits which are given in
   subroutines side-on and reflect

------------------------------------------------------------------------
int = bldgchk (to_screen, lun, fatal_error, total_fatal)
function:    to check wall area connectivity to roof and to place
             the wall index number (from column 14 of wall_table)
             of a roof area which is connected to the roofcon
             vector. this vector will have same order as wall_table
             and it is incorporated into the positions 8 & 9 of
             column 15 of wall_table. the roof connectivity is used
             in bdamawri.for to get the 'building node'
```

```
                    required by the bdama executable.
      parameters:
                    to_screen       - logical variable indicates where
                                      error messages will be displayed
                                      t : display to screen
                                      f : send to file indicated by logical
                                          unit lun
                    lun             - logical unit number for error output
                       file
                    fatal_error     - logical variable indicating if fatal
                       error occurred.  this variable is initialized once
                       at the beginning of the validation process to false
                       and set to true each time a fatal error occurs
                         f - no error occurred
                         t - fatal error occurred
                    total_fatal     - total number of fatal errors
                    the function name returns the following:
                    bldgchk : 0 - if all wall areas are connected to
                                  at least one roof area
                              1 - if this is not the case
      ------------------------------------------------------------------------
      call  bload ()
      function:     provides a menu for user selection of viewing charge
                    weight and its location or viewing the blast load on
                    building components.  data is stored in damage_table
                    as follows:
                    non-frame components:
                    col. 1 - component type in abbreviated form
                    col. 2 - peak blast pressure to component
                    col. 3 - peak blast impulse to component
                    col. 4 - local x1 coordinate of component
                    col. 5 - local x2 coordinate of component
                    col. 6 - local y1 coordinate of component
                    col. 7 - local y2 coordinate of component
                    col. 8 - distance of component from local (0,0);
                             used for sorting the damage table (program
                             use only)
                    frame components:
                    col. 1 - component type in abbreviated form
                    col. 2 - peak blast pressure to component
                    col. 3 - peak blast impulse to component
                    col. 4 - blastward wall name
                    col. 5 - local x1 coordinate of component
                    col. 6 - local x2 coordinate of component
                    col. 7 - distance of component from local (0,0);
                             used for sorting the damage table (program
                             use only)
      parameters:
      ------------------------------------------------------------------------
      call  blstrep (wall_name)
      function:     prints report on blast load for the specified wall.
                    one report is generated.  the report contains the
                    the following information depending on if the wall/roof
                    area is a frame:

                       non-frame blast load report
                          component type
                          blast load peak pressure
                          blast load impulse
                          x1 local end or opposite corner point
                          y1 local end or opposite corner point
                          x2 local end or opposite corner point
                          y2 local end or opposite corner point
```

```
                        frame blast load report
                           component type
                           blast load peak pressure
                           blast load impulse
                           blastward wall name
                           xl local end or opposite corner point
                           yl local end or opposite corner point
      ------------------------------------------------------------------------
      call  select (inbuf, trow, tcol, nrow, numcol, fldwid, -
             numval, option, option_no, init_menu, term_key, isort)
      business systems integration
      (512)680-3940
      copyright (c) 1989, 1989, 1990, 1991, 1992
      all rights reserved
      ------------------------------------------------------------------------
      call  setmsk (mask, type, m_len)
      business systems integration
      (512)680-3940
      copyright (c) 1992
      all rights reserved
      ------------------------------------------------------------------------
      call  showme (funct, row, col, wide, numopt, optl, iopt, -
             init_menu, term_key)
      business systems integration
      (512)680-3940
      copyright (c) 1988, 1989, 1990, 1991, 1992
      all rights reserved
      ------------------------------------------------------------------------
      call  spread (trow, tcol, inbuf, rowhdg, rowlab, colhdg,    -
             numrow, numcol, fldwid, tit, dsprow, dspcol, colwid,       -
             coltyp, icontinue, retopt, edt, maxhead, help_text,        -
             spread_table, spread_no, max_rows, max_cols, dup)
                           business systems integration
                                  (512) 680-3940
                              copyright (c) 1988-1992
                                all rights reserved

      type:          subroutine
      function:      permits input of data in a rectangular format
                     similar to a spreadsheet
      parameters:    trow       top row of data
                     tcol       top (left) column of data
                     inbuf      buffer of items from which the choice is made
                     rowhdg     header for row labels
                     rowlab     labels for rows
                     colhdg     labels for columns
                     numrow     number of rows in the spreadsheet
                     numcol     number of columns in spreadsheet
                     fldwid     width of each field in the buffer
                     tit        title
                     dsprow     number of rows displayed
                     dspcol     number of columns displayed
                     edt        list of rows in which an edit occurred
                     maxhead    number of rows used in column header (colhdg)
                     help_text  line of help text for each column
                     spread_no  programmer assigned spreadsheet number
                     max_rows   maximum number of rows in dimension
                     max_cols   maximum number of columns in dimension
      ------------------------------------------------------------------------
      call  buildgeo ()
      function:    allows user definition of the following:
                     1. building wall/roof area
                     2. components geometry definition
                     3. dependencies
```

```
parameters:  none
-------------------------------------------------------------------------
call  buildrep ()
function:    prints report on building damage
parameters:  none
-------------------------------------------------------------------------
call  calcdefs (calc_type, row_position, inbuf, max_rows,    -
       chosen_value, error_num)
function:    this routine calculates the a value using the eqution
             based on calc_type.  the value is converted to character
             form and returned in chosen value.  this routine is
             called by spreadop to calculate the default value in
             a field on a spreadsheet.  error checking is performed
             to test for a blank field or a field containing a 0.
parameters:
             calc_type     - contains the number of the equation
                             to be used in the calculation
             row_position  - current row on spreadsheet
             inbuf         - character array containing spreadsheet
                             values
             max_rows      - maximum row in array used to store
                             spreadsheet data, not necessarily the
                             number of rows displayed on the
                             spreadsheet.  there could be less rows
                             displayed
             chosen_value  - returns values selected by user or
                             calculated by program
             error_num     - error flag indicator
                             0 : no error occurred
                             1 : error, field was blank
                             2 : error, field was zero
-------------------------------------------------------------------------
call  cdepend (wall_name, wall_index)
function:    allows user computation and editing of dependenices for
             the specified wall/roof area.  user selects/enters from
             a list the following for each component in depend_table:
             column 1 : component type - i.e. concrete slab, concrete
                        beam, etc. (display only)
             column 2 : local x1 coordinates of end opposite corner
                        points of the component (display only)
             column 3 : local y1 coordinates of end opposite corner
                        points of the component (display only)
             column 4 : local x2 coordinates of end opposite corner
                        points of the component (display only)
             column 5 : local y2 coordinates of end opposite corner
                        points of the component (display only)
             column 6 : dependent component id number (display only)
             column 7-10 :
                        supporting component number(s) - up to 4
                        component numbers which will be pre-set
                        but may be edited by the user.
parameters:
             wall_name    - name of wall/roof area user selected
                for component definition
             wall_index   - index into the wall area used to
                access wall number for current wall_name
-------------------------------------------------------------------------
call  center(x1,x2,x3,x4,xc)

 last modified --
  6/1/90 (map)

 purpose --
  determines the center of an element
```

```
 257:
 258:        method --
 259:         see comments in code
 260:
 261:        input --
 262:         arguments:
 263:           x1,x2,x3,x4   coordinates for element end points
 264:
 265:        output --
 266:         arguments:
 267:           xc            coordinates for element center point
 268:
 269:        restrictions
 270:         2-way elements must be approximately rectangular
 271:
 272:       ------------------------------------------------------------------------
 273:       call  centerln (in_line, out_line, line_length)
 274:       function:    centers the line, in_line with a line that line_length
 275:                    long and places it centered in out_line.  the routine
 276:                    may be called with the same variable for in_line and
 277:                    out_line.
 278:       ------------------------------------------------------------------------
 279:       call  charge (ier)
 280:       function:    allows user input and editing of the load definition.
 281:                    the load definition consists of the charge weight in
 282:                    lbs and the charge global coordinates in feet.  the
 283:                    form uses load_buf to store character data.
 284:                    one form is used and its input values are as follows:
 285:                      loaddef  - form 2
 286:                                 charge weight (ft)
 287:                                 charge location in global x direction
 288:                                 charge location in global y direction
 289:                                 charge location in global z direction
 290:       parameters:
 291:                    ier          -  0 : no errors occurred
 292:                                    1 : error occurred converting data from
 293:                                        character to real.  field was not
 294:                                        entered
 295:       ------------------------------------------------------------------------
 296:       call  chgdir1 ( ier, directory_name, directory_path )
 297:                 business systems integration copyright (c) 1990
 298:                 all rights reserved
 299:
 300:       function:    change or make directory
 301:       parameters:
 302:       ------------------------------------------------------------------------
 303:       call  chgloc(xchg,xc,vn,r,angle)
 304:
 305:        last modified --
 306:         6/4/90 (map)
 307:
 308:        purpose --
 309:         calculate distance to charge and angle of blast wave impact
 310:
 311:        method --
 312:         see comments in code
 313:       subroutine child (prog, commd, icls)
 314:       ------------------------------------------------------------------------
 315:       call  chk2way (to_screen, row, lun, fatal_error,              -
 316:              error_occurred, total_fatal, total_warnings)
 317:       function:    perform validity of 2-way reinforced masonry element
 318:                    that were previously done in bdama subroutine mar2wi.
 319:                    the following items are checked:
 320:                      1. endpoints are diagonal.
```

```
                    2. aspect ratio is within reason.
      parameters:
                  to_screen      : logical variable indicating where
                                   error messages go
                                     t - messages will go to screen
                                     f - message go to file
                  row            : row number in compg of 2-w reinforced
                                   masonry element to be checked
                  lun            : logical unit number of file messages
                                   are written to if to_screen if false
                  fatal_error    : logical variable indicating if fatal
                                   error occurred.  this variable is
                                   initialized once at the beginning of
                                   the validation process to false
                                   and set to true each time a fatal
                                   error occurs
                                     f - no error occurred
                                     t - fatal error occurred
                  total_fatal    : total number of fatal errors
                  total_warnings : total number of warnings

      ------------------------------------------------------------------------
      call  chkcomp (wall_name, row, mat_name, comp_type_name,    -
             comp_prop_name, component_id, to_screen,     -
             lun, fatal_error, total_fatal,               -
             total_warnings, ier)
      function:    checks if the specified component type, comp_type_name,
                   is valid for the given material name, mat_name.  the
                   component property name is then check to make sure it
                   is valid for the specified component type.  this
                   routine is used by the preprocessor, bdamprep, and
                   the validation program, validfil.  if used by the
                   preprocessor error message go to the screen.  if used
                   by the validation program, error messages go to the
                   file specified by lun.  the following checks are made:
                     1. component type defined
                     2. component property name defined
                     3. material-component type mismatch
                     4. 1-way slabs and panels, beams and joists and
                        exterior columns are checked to insure the component
                        is linear.  this is a fatal condition in validfil.
                     5. metal stud walls and masonry 1-way components are
                        checked to see if the component is contained on
                        a wall area.  a warning is issued if it is not.
                     6. metal steel joists are checked for the component
                        being on a roof.  a warning is issued if it is not.
                     7. exterior columns and wood walls are checked to
                        ascertain if they are on a wall.  a fatal error
                        in validfil occcurs if this is not the case.
                     8. wood roofs are check to make sure the component
                        is on a roof area.  this is a fatal condition for
                        validfil.
                     9. two-way components are checked for diagonal
                        endpoints and reasonable aspect ratio.  failure
                        of one of these 2 criteria results in a fatal
                        condition.
                    10. two-way masonry components are tested for being
                        on a roof.  if this component is on a roof a
                        warning is issued.
                    11. interior columns are checked make sure end 2
                        coordinates are blank.  the coordinates are
                        blanked by the program if they contain values.
                    12. interior columns are checked to insure that end 1
                        is contained on a roof area.  a fatal error results
```

```
                        if end 1 is on a wall.
                     13. frame components are tested to makes sure the
                        blastward area is a wall.  a roof area results
                        in a fatal error.
                     14. component type-component property mismatch
                     15. component in component geometry found in
                        component properties
      parameters:
                  wall_name       : name of current wall
                  row             : current row in compg that is being
                                    processed
                  mat_name        : name of material type
                  comp_type_name  : name of component type
                  comp_prop_name  : component property name
                  component_id    : 9 character id number for component
                                    defined by the specified mat_name,
                                    comp_type_name, and comp_prop_name
                  to_screen       : logical variable indicating where
                                    error messages go
                                      t - messages will go to screen
                                      f - message go to file
                  lun             : logical unit number of file messages
                                    are written to if to_screen if false
                  fatal_error     : logical variable indicating if fatal
                                    error occurred.  this variable is
                                    initialized once at the beginning of
                                    the validation process to false
                                    and set to true each time a fatal
                                    error occurs
                                      f - no error occurred
                                      t - fatal error occurred
                  total_fatal     : total number of fatal errors
                  total_warnings  : total number of warnings
                  ier             : error flag
                                      0 - no error occurred
                                      1 - material not found
                                      2 - material-component type mismatch
                                      3 - component property name not
                                          defined
                                      4 - component type-component
                                          property mismatch
                                      5 - no component properties defined
                                          for component type
                                      6 - component type not defined
                                      7 - error occurred in itype check

      ------------------------------------------------------------------
      call  chkmastr (wall_name)
      function:    checks if any previous master component have been
                   changed to unique.  if this type of change occurred
                   the user is warned and given the chance to make
                   the component a master again and thus not lose the
                   generated components associated with it.
      parameters:
                   wall_name       : name of current wall
      ------------------------------------------------------------------
      call  compd(ndepa,idepa,xb,xchg,w,nr,dam,iwrit)

       last modified --
        11/08/91 (tkb)
        11/24/92 (dds) - increase dimesion of dam columns from 6 to 8 and
                         put pbar and ibar in those columns
        12/01/92 (dds) - commented out write to screen
        01/28/93 (dds) - removed cp2, ci2 from web steel joist read of direct
```

```
                        access file; removed idc from frames; added debuggin
                        prints

      purpose --
       determines percent damage to building components

      method --
       uses p-i curves in section 5.0 of blast vulnerability guide

      input --
       arguments:
         ndepa          number of dependency pairs in array idepa
         idepa          array with pairs of independent/dependent elements
         xb             two dimensional array with building orientation
                        coordinates
         xchg           coordinates with charge location

      output --
       arguments:
         dam            array with component damage levels and
                        repair/replace factors

      restrictions
       none set in compd

      ------------------------------------------------------------------------
      call  compdef (wall_name, wall_index)
      function:    allows user input of the components which will define
                   the specified wall/roof area.  user selects from a list
                   the following for each component:
                     1.  component material type - concrete, steel,
                         masonry or wood (selection field)
                     2.  component type - i.e. concrete slab, concrete
                         beam, etc. (selection field)
                     3.  component property name - user defined component
                         which was defined in the component properties
                         definition phase. (selection field)
                     4.  component id name - based on wall index,
                         number of component property name and number
                         generated.  the generated is 00 is master and
                         a sequential number if generated.
                         (display only field)
                     5.  x1 end or opposite corner points of the component
                         (user entry)
                     6.  y1 end or opposite corner points of the component
                         (user entry)
                     7.  y1 end or opposite corner points of the component
                         (user entry)
                     8.  y1 end or opposite corner points of the component
                         (user entry)
                     9.  master component selection
                         yes - component is a master and repeated groups
                               will be generated from it
                         no  - component is unique - no components will
                               be generated from it
                         (selection field)
                    10.  center to center spacing (user entry)
                    11.  local direction
                         x   - component generation will be in the
                               positive x direction
                         y   - component generation will be in the
                               positive x direction
                         (selection field)
                    12.  number of additional repeated groups generated
```

```
                              from the master (user entry)
                      13.  though not displayed on spreadsheet, this column
                           temporarily contains the generate flag so it
                           it not lost thru deletion from table when
                           component retrieved. (program use)
                      14-17 used by program for maintaining independent
                           component id with the corresponding dependent id.
                       1. check if any master components where changed to
                          unique components while on spreadsheet.  user
                          is prompted to restored component to master
                          status or the leave as a unique component and
                          have any generated components deleted.
                       2. component type defined
                       3. component property name defined
                       4. material-component type mismatch
                       5. 1-way slabs and panels, beams and joists and
                          exterior columns are checked to insure the component
                          is linear.  this is a fatal condition in validfil.
                       6. metal stud walls and masonry 1-way components are
                          checked to see if the component is contained on
                          a wall area.  a warning is issued if it is not.
                       7. metal steel joists are checked for the component
                          being on a roof.  a warning is issued if it is not.
                       8. exterior columns and wood walls are checked to
                          ascertain if they are on a wall.  a fatal error
                          in validfil occcurs if this is not the case.
                       9. wood roofs are check to make sure the component
                          is on a roof area.  this is a fatal condition for
                          validfil.
                      10. two-way components are checked for diagonal
                          endpoints and reasonable aspect ratio.  failure
                          of one of these 2 criteria results in a fatal
                          condition.
                      11. two-way masonry components are tested for being
                          on a roof.  if this component is on a roof a
                          warning is issued.
                      12. interior columns are checked make sure end 2
                          coordinates are blank.  the coordinates are
                          blanked by the program if they contain values.
                      13. interior columns are checked to insure that end 1
                          is contained on a roof area.  a fatal error results
                          if end 1 is on a wall.
                      14. frame components are tested to makes sure the
                          blastward area is a wall.  a roof area results
                          in a fatal error.
                      15. component type-component property mismatch
                      16. component in component geometry found in
                          component properties
                      17. component coordinates are checked to make sure
                          that coordinates are not outside wall coordinates
      parameters:
                    wall_name    - name of wall/roof area user selected
                       for component definition
                    wall_index   - index into the wall area used to
                       access wall number for current wall_name
      ------------------------------------------------------------------------
      call  compok  (wall_name, row, comp_type_name, component_id,-
            lun, fatal_error, total_fatal,                        -
            total_warnings, error_occurred)
      function:    checks if the specified component has all numeric
                   values defined and if it is a master that its slaves
                   have been generated and at least one slave exists.
      parameters:
                   wall_name       : name of current wall
```

```
                    row             : current row in compg that is being
                                      processed
                    comp_type_name  : name of component type
                    component_id    : 9 character id number for component
                                      defined by the specified mat_name,
                                      comp_type_name, and comp_prop_name
                    lun             : logical unit number of file messages
                                      are written to if
                    fatal_error     : logical variable indicating if fatal
                                      error occurred.  this variable is
                                      initialized once at the beginning of
                                      the validation process to false
                                      and set to true each time a fatal
                                      error occurs
                                        f - no error occurred
                                        t - fatal error occurred
                    total_fatal     : total number of fatal errors
                    total_warnings  : total number of warnings
                    error_occurred  : logical variable indicating if a
                                      error occurred in the checking
                                      done by this subroutine
    ------------------------------------------------------------------------
    call  compprop ()
    function:    allows user selection of component type and then
                 allows the definition of the required properties for
                 that type.  the four types are concrete, steel,
                 masonry and wood.  the spreadsheet number corresponds to
                 the icnindex of the component type.
    ------------------------------------------------------------------------
    call  compr(dam)

     last modified --
      7/15/91 (jpp)
      2/5/93 (cjo) add prestressed beams

     purpose --
      determines reusablilty of building components

     method --
      in blast vulnerability guide, section 5.0, 5.1,6.0

     input --
      ?

     output --
      ?

     restrictions --
      none set in compr
    ------------------------------------------------------------------------
    call  conv4prt (comp_table, irow, itype, imat, icomp, rval)
    function:    convert all real values in the specified row in
                 comp_table from character to real and places them
                 in rval.
    parameters:
                 comp_table  : character array containing the component
                               properties for a specified component type.
                 irow        : row in comp_table containing values to
                               be converted
                 itype       : type of component property being printed
                               (1-24)
                 imat        : material type
                 icomp       : component type
                 rval        : output real array containing converted
```

```
                                real values
      function:    prints error messages for the cell default
                   calculations in spreadop.  this routine could
                   be expanded to provide more specific error messages
                   but for now only two generic message will be printed
      parameters:  none
      ------------------------------------------------------------------------
      call  cprphead (lun, itype, report, line_kt)
      function:    writes column headers for the preprocessor component
                   property reports based on itype and report and
                   increments the line counter, line_kt, based on number
                   of lines written.
                      itype            property       # reports
                        1               rcbmi             2
                        2               rc1wi             2
                        3               rc2wi             2
                        4               rceci             2
                        5               rcici             2
                        6               rcmrfi            2
                        7               rcpsi             2
                        8               stbmi             2
                        9               stmswi            1
                       10               stowji            1
                       11               stcdi             2
                       12               steci             2
                       13               stici             2
                       14               stmrfi            2
                       15               maulwi            2
                       16               mau2wi            1
                       17               mar1wi            2
                       18               mar2wi            2
                       19               mapili            2
                       20               wdwli             2
                       21               wdrfi             2
                       22               wdbmi             2
                       23               wdeci             2
                       24               wdici             2
      parameters:
                   lun     : logical unit connected to report file
                   itype   : type of header to be printed
                   report  : indicates which report to print for the
                             specified itype (1 or 2)
                   line_kt : current number of lines written to report file
      ------------------------------------------------------------------------
      call  cprpprnt (all)
      function:    allows user to select and print reports based on
                   selection of material type and component type or print
                   all component reports.
      parameters:  all   : logical variable indicating if all reports are
                           to be printed
                           t - print all reports
                           f - show menu for user selection
      ------------------------------------------------------------------------
      call  cprprep (itype, imat, icomp, report, tot_rep, comp_name)
      function:    writes component property report for specified itype
                   and report number.
                      itype            property       # reports
                        1               rcbmi             2
                        2               rc1wi             2
                        3               rc2wi             2
                        4               rceci             2
                        5               rcici             2
                        6               rcmrfi            2
                        7               rcpsi             2
```

```
                          8                stbmi           2
                          9                stmswi          1
                         10                stowji          1
                         11                stcdi           2
                         12                steci           2
                         13                stici           2
                         14                stmrfi          2
                         15                mau1wi          2
                         16                mau2wi          1
                         17                mar1wi          2
                         18                mar2wi          2
                         19                mapili          2
                         20                wdwli           2
                         21                wdrfi           2
                         22                wdbmi           2
                         23                wdeci           2
                         24                wdici           2

      parameters:
                  itype      : type of header to be printed
                  imat       : material type
                  icomp      : component type
                  report     : number of report to be printed (1 or 2)
                  tot_rep    : total reports that will be printed for
                               the current component type
                  comp_name  : component property name of current report
      ------------------------------------------------------------------------
      call  csort2d (array, nval, isortcol, numrow, numcol, idir, -
            iconvert, ier)
      business systems integration
      (512)680-3940
      copyright (c) 1988, 1989, 1990, 1991, 1992
      all rights reserved

      function:    sort a character 2 dimensional array
      parameters:  array      character array to sort
                   nval       number of values to sort
                   isortcol   column in array sort is based on
                   numrow     number of rows to sort
                   numcol     number of columns in array
                   idir       sort direction (1 ascending, -1 descending)
                   iconvert   flag indicating whether to convert to real
                              for the column specified in doing the sort
                              0: sort the column as character
                              1: convert the number to real and sort as
                                 if column is real
                   ier        0 : no errors occurred during sort
                              1 : isortcol exceeded number of columns in
                                  array
      ------------------------------------------------------------------------
      call  damagec (damaged)
      function:    provides a menu for user selection of viewing the most
                   damaged components or all components.

      parameters:
                   damaged        : logical variable which indicates if
                                    building sustained any damage
                                    t : building sustained damage
                                    f : building sustained no significant
                                        damage (i.e. no component's damage
                                        exceeded 0%)
      ------------------------------------------------------------------------
      call  delgen (mindex, id, previous)
      function:    deletes all generated components associated with
                   the specified master id

      parameters:
```

```
                    mindex    : index to where generated components begin
                                for the specified master
                    id        : id number of master component which will
                                used to find generated components to delete
                    previous  : pointer to previous row processed
      ------------------------------------------------------------------------
      call  dellink (link, free, head, tail, current, previous)
      function:   deletes the specified component indicated by current
                  from the linked list and adds it back to the free list.
      parameters:
                  link      - link list used for maintaining order in
                              a given storage area
                  free       - pointer to first free row in storage area
                  head       - pointer to first used row in storage area
                  tail       - pointer to last used row in storage area
                  current    - pointer to current row being processed
                  previous   - pointer to previous row processed
      ------------------------------------------------------------------------
      call  delwall (wall_name)
      function:   deletes all master and unique components assoicated
                  with the specified wall, wall_name, from the compg
                  storage area.
      parameters:
                  wall_name - name of current wall/roof area
      ------------------------------------------------------------------------
      call  depin(ndep,idep,nr)

       last modified --
        6/11/91 (jpp)

       purpose --
        input, check validity of dependency list.

       method --
        2-dimensional array idep contains a list of
                independent element id, dependent element id.
        the element id's are coded to contain material id and component id
        where
                element id = (10000*imat) + (1000*icomp) + id

       input --
        arguments:
         ndep          total number of dependencies
         idep          array with pairs of independent and dependent components
         nr            three dimensional array with record numbers in
                       tape15 where component data is stored.  see
                       description in bdam.

       output --
        input arguments (ndep, idep) and:

       restrictions
        none set in depin
      ------------------------------------------------------------------------
      call  depmod(ndep,idep,ndepa,idepa)

       last modified --
        11/08/91 (tkb)

       purpose --
        modify dependency array to include multiple stepped dependencies.
        for example, if the following pairs are entered:
                  1,2
                  2,3
```

```
 833:          then the pair   1,3   should be added.
 834:
 835:        method --
 836:         see comments in code
 837:
 838:        input --
 839:         arguments:
 840:          ndep          total number of dependencies entered
 841:          idep          array with pairs of independent and dependent components
 842:                        entered by user
 843:
 844:        output --
 845:         arguments:
 846:          ndepa         total number of dependencies after modification
 847:          idepa         array with all pairs of independent and dependent component
 848:
 849:        restrictions
 850:         none set in depmod
 851:       ------------------------------------------------------------------------
 852:       call  dpndprnt (wall_name)
 853:       function:    prints report on dependencies for the specified
 854:                    wall.  one report is generated.  the report contains the
 855:                    the following information.
 856:
 857:                       non-frame dependency report 1
 858:                          component type
 859:                          x1 local end or opposite corner point
 860:                          y1 local end or opposite corner point
 861:                          x2 local end or opposite corner point
 862:                          y2 local end or opposite corner point
 863:                          dependent component id #
 864:                          independent component id # (1-4)
 865:       ------------------------------------------------------------------------
 866:       call  editform (form_file_name, form_number, usrdat,          -
 867:              key_pressed)
 868:       routine    : editform
 869:       function   : displays the specified form, form_file_name, to the
 870:                    screen and allows user processing of that form
 871:       parameters : form_file_name - name of form, with full path, to be
 872:                       edited
 873:
 874:                    form_number     - number of form to be edited
 875:                    usrdat          - array containing form data
 876:                    key_pressed     - the key stroke the user left the
 877:                                      form with
 878:       ------------------------------------------------------------------------
 879:       call  editfrm2 (form_file_name, form_number, usrdat,          -
 880:              key_pressed)
 881:       routine    : edtfrm2
 882:       function   : displays the specified form, form_file_name, to the
 883:                    screen and allows user processing of that form for
 884:                    the bdam postprocessor
 885:       parameters : form_file_name - name of form, with full path, to be
 886:                       edited
 887:
 888:                    form_number     - number of form to be edited
 889:                    usrdat          - array containing form data
 890:                    key_pressed     - the key stroke the user left the
 891:                                      form with
 892:       ------------------------------------------------------------------------
 893:       call  filexist (input_file_name, ier)
 894:       routine    : filexist
 895:       function   : allows user to select the bdamprep file name to save
 896:                    the file under.  the default is the current file name.
```

```
                    issues warning message if the selected file already
                    exists.  user then has the choice of writing over the
                    existing file or aborting the save.
      parameters :
                    input_file_name    : current preprocessor file name
                                         selected by user.  contains full
                                         path
                    ier                : 0 - save file
                                       : 1 - do not save file
      ------------------------------------------------------------------------
      call  fillblst (wall_name, number_damaged)
      function:     fills the damage_table with required data for displaying
                    all components for a specified wall area.
                    data is stored in damage_table as follows:
                    non-frame components:
                    col. 1 - component type in abbreviated form
                    col. 2 - peak blast pressure to component
                    col. 3 - peak blast impulse to component
                    col. 4 - local x1 coordinate of component
                    col. 5 - local x2 coordinate of component
                    col. 6 - local y1 coordinate of component
                    col. 7 - local y2 coordinate of component
                    col. 8 - distance of component from local (0,0);
                             used for sorting the damage table (program
                             use only)
                    frame components:
                    col. 1 - component type in abbreviated form
                    col. 2 - peak blast pressure to component
                    col. 3 - peak blast impulse to component
                    col. 4 - blastward wall name
                    col. 5 - local x1 coordinate of component
                    col. 6 - local x2 coordinate of component
                    col. 7 - distance of component from local (0,0);
                             used for sorting the damage table (future
                             program use if decided to perform a secondary
                             sort)
      parameters:
                    wall_name      : current wall/roof area name
                    number_damaged : total number of most damaged components
                                     for specified wall
      ------------------------------------------------------------------------
      call  fillcdam (wall_name, number_damaged)
      function:     fills the damage_table with required data for displaying
                    components for a specified wall area.
                    data is stored in damage_table as follows:
                    non-frame components:
                    col. 1 - component type in abbreviated form
                    col. 2 - local x1 coordinate of component
                    col. 3 - local x2 coordinate of component
                    col. 4 - local y1 coordinate of component
                    col. 5 - local y2 coordinate of component
                    col. 6 - percent damage to component
                    col. 7 - repair or replace flag to component
                    col. 8 - p-i diagram terms pbar
                    col. 9 - p-i diagram terms ibar
                    col.10 - distance of component from local (0,0);
                             used for sorting the damage table (program
                             use only)
                    frame components:
                    col. 1 - component type in abbreviated form
                    col. 2 - percent damage to component
                    col. 3 - repair or replace flag to component
                    col. 4 - p-i diagram terms pbar
                    col. 5 - p-i diagram terms ibar
```

```
                    col. 6 - blastward wall name
                    col. 7 - local x1 coordinate of component
                    col. 8 - local x2 coordinate of component
                    col. 9 - distance of component from local (0,0);
                             used for sorting the damage table (future
                             program use if decided to perform a secondary
                             sort)
     parameters:
                    wall_name       : user assigned name of wall/roof area,
                                      unless the area is a frame in which
                                      case the program assigned the area name
                                      of frame
                    number_damaged  : total number of most damaged components
                                      for specified wall
     ------------------------------------------------------------------------
     call  filldp (mindex, dindex, blankit)
     function:   copies the data required for the dependency spreadsheet
                 depend_table from compg.  note: if storage of compg
                 or depend_table changes, indexes on the do loops will
                 have to change.
     parameters:
                 mindex       - row index of compg table to be filled
                 dindex       - row index of depend_table to be filled
                 blankit      - logical variabe indicating if old
                                dependencies are to be copied or blanked
                                t : blank old dependencies in compg and
                                    depend_table
                                f : copy old dependencies from compg to
                                    depend_table

     ------------------------------------------------------------------------
     call  fillgeo (wall_name, num_print, ier)
     function:   retrieve a master/unique data and generated components
                 for the specified wall_name from compg and places in
                 the work_table for printing.  for non-frame items
                 the distance from the local 0,0 to the endpoints is
                 placed in column 14 of work_table for each row.  for
                 frames the is placed in column 8.  these distances
                 will be used for sorting.
     parameters:
                 wall_name    - user assigned wall name which is
                                to be displayed with dependencies
                 num_print    - number of components put in work_table
                                to be printed
                 ier          - 0 : no errors occurred
                                1 : no componets found for wall
                                2 : number of components exceeded
                                    numrow
     ------------------------------------------------------------------------
     call  fillmdam (number_most_damaged, frames)
     function:   fills the damage_table with required data for displaying
                 most damaged components.
                 non-frame components:
                 col. 1 - wall name component is on
                 col. 2 - component type in abbreviated form
                 col. 3 - percent damage to component
                 col. 4 - local x1 coordinate of component
                 col. 5 - local x2 coordinate of component
                 col. 6 - local y1 coordinate of component
                 col. 7 - local y2 coordinate of component
                 col. 8 - p-i diagram terms pbar
                 col. 9 - p-i diagram terms ibar
                 frame components:
                 col. 1 - blastward wall name (has * in 1st column
```

```
                                to distinquish from non-frame components
                        col. 2 - component type in abbreviated form
                        col. 3 - percent damage to component
                        col. 4 - local x1 coordinate of component
                        col. 5 - local x2 coordinate of component
                        col. 6 - p-i diagram terms pbar
                        col. 7 - p-i diagram terms ibar
      parameters:
                        number_most_components : total number of most damages
                                                 components
                        frames                 : logical variable
                                                 t - frame components were among
                                                     most damaged
                                                 f - no components were among
                                                     most damaged
      ------------------------------------------------------------------------
      call  fndcomp (mat_name, comp_type_name, comp_prop_name, ier)
      function:    provides pop-up menu of list of defined component
                   property names based on the current material type
                   and component type.  the selected name is returned in
                   comp_prop_name along with an error flag.
      parameters:
                   mat_name       : name of material type
                   comp_type_name : name of component type
                   comp_prop_name : selected component property name
                   ier            : 0 - no error occurred
                                    1 - material name or component name
                                        is not defined
                                    2 - material name not found in material
                                        list - programmer error
                                    3 - component name not found in
                                        component list - programmer error
                                    4 - no component properties defined
                                        for specified material and component
                                        type
                                    5 - user aborted selection of component
                                        property name

      ------------------------------------------------------------------------
      call  fndgen (m_id, mindex, found)
      function:    searches the generated link list for where data for
                   specified master component begins.

      parameters:
                   part_id
                   mindex
                   found
      ------------------------------------------------------------------------
      call  fndwcomp (wall_name, mlink, head, mindex, previous,    -
             itype, found, id)
      function:    searches the specified link list for where data for
                   specified wall_name begins.  if type = 1, the master/
                   unique list is searched until the specified wall_name
                   if found.  the pointer, mindex then returns the row
                   number of the found record; found is set to true and
                   previous returns the row number in the list before
                   the found record.  if type = 2, the generated
                   list is also searched for the specified wall_name.  once
                   the wall is found the search switches to searching
                   for the id of the master the components where generated
                   from.  found is set to true, if data for
                   specified wall is found.  found is false if not found.
                   mindex returns the row number of the first wall_name
                   component in the list.  previous returns the row
```

```
                number in the list before the record was found.

parameters:
                wall_name       : name of current wall
                head            : pointer to head of linked list
                mindex          : pointer to where data for the specified
                                  wall begins
                previous        : pointer to previous row processed
                itype           : indicates if storing master/unique,
                                  generated or frame components
                                  1 - master/unique component
                                  2 - generated component
                                  3 - frame
                found           : logical variable that indicates
                                  finding of desired data
                                  t - components found for specified
                                      wall
                                  f - no components found for specified
                                      wall
                id              : if itype = generated, id contains
                                  the number of the master id the
                                  components were generated from
-------------------------------------------------------------------------
call  framdef ()
function:   allows user input of the frame components.  user selects
            the following items indicate the columns used in the
            work area, frame_table:
              1.  frame type - reinforced concrete frame or
                  steel frame
              2.  component property name - user defined component
                  which was defined in the component properties
                  definition phase. (selection field)
              3.  frame id - same format as component id but to
                  indicate this is a frame the first portion is
                  00.  the second number is based on a sequential
                  number assigned for the frames and last portion
                  is 00. (display only field)
              4.  wall containing frame which is nearest the blast.
                  (selection field)
              5.  x1 coordinate of ground level endpoint of column
                  (user entry)
              6.  y1 coordinate of ground level endpoint of column
                  (user entry)
              7.  total height of frame (user entry)
              8.  material type - concrete or steel, dependent on
                  component type selected (program use)
parameters:
-------------------------------------------------------------------------
call  framekil (rownum, depcomp, numdcomp)
function:   finds all components within the area of influence
            of a frame and sends the array depcomp with the id numbers
            of all these components back to the calling
            program
parmeters:
            rownum      - the row number of frame component
                          compg matrix
            depcomp     - an array with id numbers for all
                          components which are failed due
                          to failure of frame in rownum
            numdcomp    - number of component id numbers in
                          depcomp
-------------------------------------------------------------------------
call  gen (wall_index, set_gen_flag, w_comp_tmp, ier)
function:   generates the requested components from the master
```

```
                    component.
      parameters:
                    wall_index   - index into the wall area used to
                       access wall number for current wall_name
                    set_gen_flag - .true. : indicates that generated
                       components were created from the specified master
                       and that the gen_flag parameter in associated with
                       the compg storage area should be updated to reflect
                       that components were generated from that master.
                    w_comp_tmp   - temporary storage for the row containing
                       the master component that components are to be
                       generated from.  this is done since the work space,
                       work_table, is actually the same for master/unique
                       components and for the generated components.
                    ier          - 0 : no error occurred
                                   1 : one or more required pieces of data
                                       in missing from the current master
      ------------------------------------------------------------------------
      call  gendpnd (wall_name, wall_index, total_depend)
      function:    finds independent component id numbers for each component
                   this is a 'first-cut' method in that it is based on a
                   number of assumptions discussed in user's manual
      parmeters:
                   wall_name    - name of wall/roof area user selected
                      for component definition
                   wall_index   - index into the wall area used to
                      access wall number for current wall_name
                   total_depend - total number of components in current
                      wall
      ------------------------------------------------------------------------
      call  genframd ()
      function:    generated dependencies for frame.  finds all frames
                   which have a damage of 100% and then find all components
                   within the volume of the frame and sets them to
                   total failure (100%) also.  recalculate building damage,
                   building repair/replace factor and % reusable floor.
      parameters:
      ------------------------------------------------------------------------
      call  genit (wall_name, wall_index, id, num_gen, w_comp_tmp,-
            term_key, ier)
      function:    allows user to view/ edit generated components from
                   the master component .  user selects/enters from a
                   list the following for each component:
                     1.  component material type - concrete, steel,
                            masonry or wood (selection field)
                     2.  component type - i.e. concrete slab, concrete
                            beam, etc. (selection field)
                     3.  component property name - user defined component
                            which was defined in the component properties
                            definition phase. (selection field)
                     4.  component id name - based on wall index,
                            number of component property name and number
                            generated.  the generated is 00 is master and
                            a sequential number if generated.
                            (display only field)
                     5.  x1 end or opposite corner points of the component
                            (user entry)
                     6.  y1 end or opposite corner points of the component
                            (user entry)
                     7.  y1 end or opposite corner points of the component
                            (user entry)
                     8.  y1 end or opposite corner points of the component
                            (user entry)
      parameters:
```

```
                    wall_name    - name of wall/roof area user selected
                       for component definition
                    wall_index   - index into the wall area used to
                       access wall number for current wall_name
                    id           - id number of component which generated
                                   components
                    num_gen      - number of additional repeat groups generated
                                   for current master component
                    w_comp_tmp-  temporary storage for the row containing
                                   the master component that components are
                                   to be generated from.  this is done since
                                   the work space, work_table, is actually
                                   the same for master/unique components and
                                   for the generated components.
                    term_key     - key_stroke user exited on
                    ier          - 0 : no error occurred
                                   1 : error occurred opening sprdhead.dat
                                       or reading file
    ------------------------------------------------------------------------
    call  geomprnt (wall_name)
    function:    prints report on component geometry for the specified
                 wall.  the report contains the following information
                 depending on if the wall/roof area is a frame:

                    non-frame component geometry report 1
                       item #
                       material type
                       component type
                       component property name
                       component id #

                    non-frame component geometry report 2
                       x1 local end or opposite corner point
                       y1 local end or opposite corner point
                       x2 local end or opposite corner point
                       y2 local end or opposite corner point
                       master component generation (yes or no)
                       center to center spacing
                       local direction of c/c spacing (x or y)
                       number of components to be generated

                    frame component geometry report
                       component type
                       component property name
                       component id #
                       blastward wall name
                       x1 local end coordinate
                       y1 local end coordinate
                       height of frame
    ------------------------------------------------------------------------
    call  getdata (input_file_name)
    function:    retrieve data for current problem which is required
                 for a restart of the preprocessor.
    parameters:
                 input_file_name    : name of file to retrive data
                                      from; contains full path
    ------------------------------------------------------------------------
    call  getdpnd (wall_name, numrow, blankit, total_dpnd, ier)
    function:    retrieve a master/unique data and generated components
                 for the specified wall_name from compg and places in
                 the depend_table for spreadsheet display.
    parameters:
                 wall_name    - user assigned wall name which is
                                to be displayed with dependencies
```

```
                    numrow      - maximum number of rows which can be
                                  displayed on spreadsheet
                    blankit     - logical variabe indicating if old
                                  dependencies are to be copied or blanked
                                  t : blank old dependencies in compg and
                                      depend_table
                                  f : copy old dependencies from compg to
                                      depend_table
                    total_dpnd  - total number of components for current
                                  wall/roof area
                    ier         - 0 : no errors occurred
                                  1 : no componets found for wall
                                  2 : number of components exceeded
                                      numrow
      ------------------------------------------------------------------------
      call  getinp (row, pcol, prompt, lprmpt, col, mtype, mlen, -
            outstr, term, clegal, numter)
      business systems integration
      (512)680-3940
      copyright (c) 1988, 1989, 1990, 1991, 1992
      all rights reserved
      ------------------------------------------------------------------------
      call  getspred (spread_no, ier)
      function:   reads all required variables for the specified
                  spreadsheets from the data file sprdhead.dat.
                  the following values are read:
                  1. spread no
                  2. number of columns on spreadsheet
                  3. number of rows on spreadsheet
                  4. number of column header lines
                  5. column headers
                  6. help text
                  7. variable type for each column (i.e. character,
                     integer, real)
                  8. number of defaults
                  9. column numbers of columns which have default settings
                 10. default value for the default columns
                 11. fields where duplication on spreadsheet allowed
                     0 - field can be duplicated
                     1 - field can not be duplicated
      parameters:
                  spread_no   - number of spreadsheet to retrieve data for
                  ier         - 0 : no error occurred
                                1 : error occurred opening file,
                                    sprdhead.dat
                                2 : error occurred reading sprdhead.dat
      ------------------------------------------------------------------------
      call  getvars (spread_no, col_position, calc_type)
      function:   used by spreadop for determining the columns used in
                  default calculations.  based on the spreadsheet number,
                  spread_no and the column position on the spreadsheet,
                  col_position, the required column numbers are returned
                  along with an indicator of the equation to be used for
                  calcluating.
      ------------------------------------------------------------------------
      call  getwalls (wall_name, term_key, ier)
      function:   provides pop-up menu of list of defined wall areas
                  (roofs are excluded) for user selection.  the selected
                  name is returned along with the keystroke the user
                  exited the menu with.
      parameters:
                  wall_name   : name of chosen wall/roof area
                  term_key    : reflects user's decision to continue
                                processing or return to previous menu
```

```
                                        0 - continue processing
                                        f2 (316) - return to previous menu
                    ier               : 0 - no error occurred
                                        1 - no wall areas defined
                                        2 - or user aborted selection

    ------------------------------------------------------------------------
    call  getwname (wall_name, wall_index, itype, term_key, ier)
    function:     provides pop-up menu of list of defined wall/roof
                  areas.  the selected name is returned along with
                  the keystroke the user exited the menu with.
    parameters:
                  wall_name       : name of chosen wall/roof area
                  wall_index      : index into wall data for wall/roof
                                    area selected
                  itype           : indicator of whether to display message
                                    applying to wall component definitions
                                    or dependencies
                                    1 - component definitions
                                    2 - dependencies
                                    3 - blast load on building components
                                    4 - most damaged components
                                    5 - component geometry print
                                    6 - dependency print
                  term_key        : reflects user's decision to continue
                                    processing or return to previous menu
                                    0 - continue processing
                                    f2 (316) - return to previous menu
                  ier             : 0 - no error occurred
                                    1 - no wall areas defined or user
                                        aborted selection

    ------------------------------------------------------------------------
    call  gtcompgr (comp_id, irow, ier)
    function:     uses the specified component id to determine the
                  row number of the component property in compg storage
                  area.
    parameters:
                  comp_id           : component id number used for searching
                                      the compg array
                  irow              : row number of component in compg area
                  ier               : 0 - no error occurred
                                      1 - not found in compg table

    ------------------------------------------------------------------------
    call  gtfulnam (comp_type_name, icomp, imat, ier)
    function:     uses the specified component property name to find
                  the icomp and imat that corresponds to the specified
                  type.
    parameters:
                  comp_type_name : name of component type
                  icomp          : component # with a given material group
                  imat           : material #
                  ier            : 0 - no error occurred
                                   1 - component type name not found
    ------------------------------------------------------------------------
    call  gtgroup (wall_name, itype, id)
    function:     retrieves the wall/roof roof group specified by wall name
                  of master and unique components, if itype = 1,  from
                  the compg storage array and places them in w_comp_table
                  for spreadsheet display.  the generate flag is
                  maintained aligned with the correct component by
                  removing it from compg and place it in an unused col,
                  column 13, of the w_comp_table.  if itype = 2, generated
```

```
1409:                    components are being retrieved from compg and places
1410:                    in gencomp.  if itype = 3, frame componets are being
1411:                    retrieved from compg and placed in frame table.
1412:                    compg contains all required information for each
1413:                    component used in the building.  the rows within compg
1414:                    are never physically moved but order is maintained
1415:                    thru the link list mlink.  pointers to the beginning
1416:                    and end of the used master/unique components are
1417:                    maintained in head and tail.  pointers to the
1418:                    beginning and end of the used generated nodes are
1419:                    maintained in ghead and gtail.  pointers to the
1420:                    beginning and end of the frame nodes are maintained in
1421:                    fhead and gtail.
1422:
1423:                    compg contains the following for itypes 1 & 2:
1424:                    non-frame components:
1425:                    column 1 : wall name
1426:                    column 2 : component material type (wood, concrete, etc)
1427:                    column 3 : component type (r/c beam, etc)
1428:                    column 4 : component property name (user assigned)
1429:                    column 5 : component id number
1430:                    column 6 : x1 coordinates of end or corner
1431:                    column 7 : y1 coordinates of end or corner
1432:                    column 8 : x2 coordinates of end or corner
1433:                    column 9 : y2 coordinates of end or corner
1434:                    column 10: master component indicator
1435:                               yes - master
1436:                               no  - unique
1437:                    column 11: center to center spacing
1438:                    column 12: local direction
1439:                               local x - generate in positive x direction
1440:                               local y - generate in positive y direction
1441:                    column 13: number of additional repeated components
1442:                    column 14: independent component 1
1443:                    column 15: independent component 2
1444:                    column 16: independent component 3
1445:                    column 17: independent component 4
1446:
1447:                    compg contains the following for itype 3:
1448:                    frame components
1449:                    column 1 : wall name - always frame
1450:                    column 2 : component material type (concrete or steel)
1451:                    column 3 : component type (r/c frame or steel frame)
1452:                    column 4 : component property name (user assigned)
1453:                    column 5 : component id number
1454:                    column 6 : x1 coordinate of blastward exterior column,
1455:                               coordinates are in the local coordinates of
1456:                               the blastward wall
1457:                    column 7 : y1 coordinate of blastward exterior column,
1458:                               coordinates are in the local coordinates of
1459:                               the blastward wall
1460:                    column 8 : blank
1461:                    column 9 : blank
1462:                    column 10: set to no since no components may be generated
1463:                               from this component
1464:                    column 11: name of blastward wall
1465:                    column 12: total frame height
1466:
1467:          parameters:
1468:                    wall_name - name of current wall/roof area
1469:                    itype     - indicates if storing master/unique,
1470:                                generated or frame components
1471:                                1 : master/unique component
1472:                                2 : generated component
```

```
                                3 : frame components
                id          - id number of master component which will
                              used to find components generated from
                              that master (used when itype = 2)
     ------------------------------------------------------------------------
     call  gtmat (imat, icomp)
     function:    retrieves the component material property specified
                  by imat and icomp from the xmp storage array and places
                  them in comp_table for spreadsheet display.  xmp
                  contains all required information for component
                  material property used in the building.  the rows
                  within xmp are never physically moved but order is
                  maintained thru the link list clink.

                  xmp contains the following:
                  column 1 : icn - identification based on material
                                   and selected subset component
                  column 2 : component property name (user assigned)
                  column 3 : weighting factor
                  column 4 : beam width
                  column 5 : beam thickness
                  column 6 : loaded width
                  column 7 : total weight
                  column 8 : compressive concrete strength
                  column 9 : steel yield strength
                  column 10: depth to tensile steel
                  column 11: area tensile steel
                  column 12: moment of inertia
                  column 13: boundary condition
     parameters:
                  imat       - material indicator
                               1 : concrete
                               2 : steel
                               3 : masonry
                               4 : wood
                  icomp      - number of subcategory which occurs under
                               each material
     ------------------------------------------------------------------------
     call  gtsortid (compg_row, sort_row, ier)
     function:    uses the specified row number in compg to search
                  the sortcompg table to determine the index into the
                  damage table, dam, from bdama.
     parameters:
                  comp_row        : row number of current component in
                                    the compg array
                  sort_row        : row number of component in sortcompg
                                    and dam
                  ier             : 0 - no error occurred
                                    1 - not found in sortcompg table

     ------------------------------------------------------------------------
     call  gtxmprow (mat_name, comp_type_name, comp_prop_name,   -
           irow, ier)
     function:    uses the specified material type, mat_name, the
                  specified component type, comp_type_name, and the
                  specified component property name to determine the
                  row number of the component property in xmp storage
                  area.
     parameters:
                  mat_name        : name of material type
                  comp_type_name  : name of component type
                  comp_prop_name  : name of component property name
                  irow            : row number of component in xmp area
                  ier             : 0 - no error occurred
```

```
                                        1 - invalid material type
                                        2 - specified component type not
                                            valid for specified material
                                        3 - specified component property not
                                            valid for specified component type

      ------------------------------------------------------------------------
      call  inslink (link, free, head, tail, current, ier)
      function:    removes the first available free space from the free
                   list and inserts it to the tail of the linked list.
      parameters:
                   link       - link list used for maintaining order in
                                a given storage area
                   free       - pointer to first free row in storage area
                   head       - pointer to first used row in storage area
                   tail       - pointer to last used row in storage area
                   current    - pointer to current row being processed
                   ier        - 0 : no error occurred
                                1 : no more free space available
                                    in storage area - it's full
      ------------------------------------------------------------------------
      call  interp(imat,icomp,pbar,ibar,dmg,n)

        last modified --
          10/28/91   (tkb)
          2/8/93  (cjo) asymptotes for all components added into interp
                  new asymptotes added for components steel components
                  without tension membrane and rc components with arching
                  wood comp. asymptotes changed to correlate to new p-i
                  parameters

        purpose --
          determination of percent damage to component to be used in level
          of damage evaluation

        method --
          numerically defines relevant p-i curves for material/component
          type under consideration, and compares these values with calculated
          pbar and ibar terms to identify the corresponding level of damage.

        input --
          imat
          icomp
          pbar
          ibar

        output --
          dmg

        restrictions --
          none set in interp
      ------------------------------------------------------------------------
      int = iroof (wall_index)
      function:    to determine if wall area in row 'wall_index'
                   is a 'roof'.  roof returns a 1 if the area is a roof
                   otherwise it returns a 0.
      parameters:
                   wall_index   - index into the wall area used to
                      access wall number for current wall_name
      ------------------------------------------------------------------------
      call  itypechk (to_screen, row, wall_name, comp_type,       -
             comp_id, imat, icomp, lun, fatal_error,     -
             error_occurred, total_fatal, total_warnings)
      function:    checks for component specific errors based on the itype
```

```
1601:                    of the component.  the following conditions are checked:
1602:                      1. 1-way slabs and panels, beams and joists and
1603:                         exterior columns are checked to insure the component
1604:                         is linear.  a warning is issued if it is not.
1605:                      2. metal stud walls and masonry 1-way components are
1606:                         checked to see if the component is contained on
1607:                         a wall area.  a warning is issued if it is not.
1608:                      3. metal steel joists are checked for the component
1609:                         being on a roof.  a warning is issued if it is not.
1610:                      4. exterior columns and wood walls are checked to
1611:                         ascertain if they are on a wall.  a fatal error
1612:                         in validfil occcurs if this is not the case.
1613:                      5. wood roofs are check to make sure the component
1614:                         is on a roof area.  this is a fatal condition for
1615:                         validfil.
1616:                      6. two-way components are checked for diagonal
1617:                         endpoints and a minimum aspect ratio.  failure
1618:                         of one of these 2 criteria results in a fatal
1619:                         condition.
1620:                      7. two-way masonry components are tested for being
1621:                         on a roof.  if this component is on a roof a
1622:                         warning is issued.
1623:                      8. interior columns are checked make sure end 2
1624:                         coordinates are blank.  the coordinates are
1625:                         blanked by the program if they contain values.
1626:                      9. interior columns are checked to insure that end 1
1627:                         is contained on a roof area.  a fatal error results
1628:                         if end 1 is on a wall.
1629:                     10. frame components are tested to makes sure the
1630:                         blastward area is a wall.  a roof area results
1631:                         in a fatal error.
1632:          parameters:
1633:                    to_screen      : logical variable indicating where
1634:                                     error messages go
1635:                                       t - messages will go to screen
1636:                                       f - message go to file
1637:                    row            : current row in compg that is being
1638:                                     processed
1639:                    wall_name      : name of wall/roof area user selected
1640:                                     for component definition
1641:                    comp_type      : name of component type
1642:                    comp_id        : 9 character id number for component
1643:                                     defined by the specified mat_name,
1644:                                     comp_type_name, and comp_prop_name
1645:                    imat           : number between 1 and 4 which indicates
1646:                                     the material type of the current
1647:                                     component
1648:                    icomp          : number between 1 and 7 which indicates
1649:                                     the current component number in the
1650:                                     given material group
1651:                    lun            : logical unit number for error output
1652:                                     file
1653:                    fatal_error    : logical variable indicating if fatal
1654:                                     error occurred.  this variable is
1655:                                     initialized once at the beginning of
1656:                                     the validation process to false and
1657:                                     set to true each time a fatal error
1658:                                     occurs
1659:                                       f - no error occurred
1660:                                       t - fatal error occurred
1661:                    error_occurred : indicates error occurred in this
1662:                                     category
1663:                    total_fatal    : total number of fatal errors which have
1664:                                     occurred
```

```
1665:                    total_warnings : total number of warnings
1666:       ------------------------------------------------------------------
1667:       call  ktdpnd (total_depend)
1668:       function:    checks the compg area to count the total number of
1669:                    dependencies pairs that exist.
1670:       ------------------------------------------------------------------
1671:       call  ktwdpnd (wall_name, total_depend)
1672:       function:    checks the compg area to count the total number of
1673:                    independent components that exist for the current wall
1674:       ------------------------------------------------------------------
1675:       call  leftj (chosen_value, char_variable, variable_length)
1676:       function:    left justifies data within a variable.  char_variable
1677:                    contains the data to be right justified.  chosen_value
1678:                    contains the left justified data.  variable should
1679:                    not be longer than 80 characters
1680:       ------------------------------------------------------------------
1681:       call  mapili (nc1,nel)
1682:
1683:        masonry pilasters
1684:
1685:        last modified --
1686:         7/01/91 (jpp)
1687:        01/28/93 (dds) - made moment of inertia be input, xim, rather than
1688:                         calculated value, im; added debugging prints
1689:
1690:        purpose --
1691:         data is read from and/or written to a direct access file.
1692:
1693:        method --
1694:         in blast vulnerability guide, see section 5.0, figure 5.1*
1695:         and section 6.1                                          *
1696:
1697:        input --
1698:         tape15:  direct access file with individual element data.
1699:
1700:        output --
1701:         tape15 and arguments (see input discussion)
1702:
1703:        restrictions
1704:         none set in mapili
1705:       ------------------------------------------------------------------
1706:       call  marlwi (nc1,nel)
1707:
1708:        reinforced masonry one way walls
1709:
1710:        last modified --
1711:         7/01/91 (jpp)
1712:        01/28/93 (dds) - made moment of inertia be input, xim, rather than
1713:                         calculated value, im; removed a from read and write;
1714:                         added debugging prints, weight/length in ci is
1715:                         calculated using areal weight and section width
1716:
1717:        purpose --
1718:         data is read from and/or written to a direct access file.
1719:
1720:        method --
1721:         in blast vulnerability guide, see section 5.0, figure 5.1*
1722:         and section 6.1                                          *
1723:
1724:        input --
1725:         tape15:  direct access file with individual element data.
1726:
1727:        output --
1728:         tape15 and arguments (see input discussion)
```

```
1729:
1730:        restrictions
1731:         none set in mar1wi
1732:       ------------------------------------------------------------------
1733:       call  mar2wi(nc1,nel)
1734:
1735:        reinforced masonry two way walls
1736:
1737:        last modified --
1738:         7/02/91 (jpp)
1739:        01/28/93 (dds) - made moment of inertia be input, xim, rather than
1740:                         calculated value, im; added debugging prints
1741:                         modified gamma in ci into effective wall density
1742:
1743:        purpose --
1744:         data is read from and/or written to a direct access file.
1745:
1746:        method --
1747:         in blast vulnerability guide, see section 5.0, figure 5.1*
1748:         and section 6.1                                          *
1749:
1750:        input --
1751:         tape15:  direct access file with individual element data.
1752:
1753:        output --
1754:         tape15 and arguments (see input discussion)
1755:
1756:        restrictions
1757:         none set in mar2wi
1758:       ------------------------------------------------------------------
1759:       call  maulwi (nc1,nel)
1760:
1761:        unreinforced masonry one way walls
1762:
1763:        last modified --
1764:         7/03/91 (jpp)
1765:        01/28/93 (dds) - made moment of inertia be input, xim, rather than
1766:                         calculated value im; made z input rather than calculated
1767:                         value; added iarch; and added debugging prints; removed
1768:                         cross-sectional area, a, from input list
1769:                         modified ci to get weight/length in terms of gamma
1770:
1771:        purpose --
1772:         data is read from and/or written to a direct access file.
1773:
1774:        method --
1775:         see theory manual
1776:
1777:        input --
1778:         tape15:  direct access file with individual element data.
1779:
1780:        output --
1781:         tape15 and arguments (see input discussion)
1782:
1783:        restrictions
1784:         none set in maulwi
1785:       ------------------------------------------------------------------
1786:       call  mau2wi(nc1,nel)
1787:
1788:        unreinforced masonry two way walls
1789:
1790:        last modified --
1791:         7/03/91 (jpp)
1792:        01/28/93 (dds) - added b to read, write; added debugging prints
```

```
                        modified ci to calculate density with gamma

      purpose --
       data is read from and/or written to a direct access file.

      method --
       in blast vulnerability guide, see section 5.0, figure 5.1*
       and section 6.1                                          *

      input --
       tape15:  direct access file with individual element data.

      output --
       tape15 and arguments (see input discussion)

      restrictions
       none set in mau2wi
     ------------------------------------------------------------------
     call  mdamrep ()
     function:    prints report on the most damaged components.  the
                  report contains the information as follows:
                    most damaged building components report  -
                       wall/roof name
                       component type
                       percent damage
                       component local coordinates
                       p-i diagram terms, pbar
                       p-i diagram terms, ibar
     parameters:  none
     ------------------------------------------------------------------
     call  onorm(xc,x1,x2,xb,vn)

      last modified --
       6/4/90 (map)

      purpose --
       calculate outward normal vector from center of component

      method --
       see comments in code

      input --
       arguments:
        xc            coordinates of center of component
        x1,x2         coordinates of two nodes on element which are not
                      colinear with xc
        xb            coordinates of building orientation node

      output --
       arguments:
        vn            components of unit outward normal vector

      restrictions
       none set in onorm
     ------------------------------------------------------------------
     call  optchk (option_table, key_pressed, match, old_key)
     function:     check if keypressed was option key or if mouse was
                   clicked within the range of the option field

     parameters:   inp     option_table - table defining option area
                              where mouse can be clicked and what key
                              that click translates to.
                              1  form number
                              2  row number of option
```

```
                                3  column number of option
                                4  width of option field
                                5  key which that area represents
                                6  1: regular option
                                   2: external option field

                    inp     key_pressed - key_pressed by user
                    out     key_pressed - if was mouse click and within
                               option field, key_pressed set to key
                               represented by the click
                    out     match
                               true :  click or key pressed within option
                                       range
                               false:  click or key pressed not with
                                       option range

    ------------------------------------------------------------------------
    call  optionb (usrdat, key_pressed, form_number, ipos)
    function:    provides the options for forms in bdampost.  however,
                 since no external options are used by the current
                 forms in bdampost, this is a dummy routine and only a
                 skeleton structure set up in case it might later be
                 needed.
    parameters:
                 usrdat          : array containing form data
                 key_pressed     : keystroke user left form with and
                                   is used as an input parameter into
                                   this routine to determine action to
                                   take
                 form_number     : number of form being edited
                 ipos            : current row number in form (usrdat)
                                   when entered this routine
    ------------------------------------------------------------------------
    call  optionb2 (usrdat, key_pressed, form_number, ipos)
    function:    dummy program for editfrm2 since facedap forms have no
                 external options
    ------------------------------------------------------------------------
    call  out1(name,w,xchg,ninc,dxchg,xb,nbop,ndep,idep,nr)

     last modified --
       7/10/91    (map,jpp)

     purpose --
       writes input data to output file

     method --


     input --


     output --


     restrictions --
       none set in out1
    ------------------------------------------------------------------------
    call  out2(w,xchg,dam,bd,br,bu,icount)

     last modified --
       7/10/91    (map,jpp)

     purpose --
       writes result to output file
```

```
1921:
1922:        method --
1923:
1924:
1925:        input --
1926:
1927:
1928:        output --
1929:
1930:
1931:        restrictions --
1932:          none set in out2
1933:       ------------------------------------------------------------------
1934:       call  outpl1(name,w,ninc)
1935:
1936:        last modified --
1937:          7/30/91    (jpp)
1938:
1939:        purpose --
1940:          writes title, charge weight, and number of iterations to
1941:          output plot file
1942:       ------------------------------------------------------------------
1943:       call  outpl2(xb,xchg,bd,br,bu)
1944:
1945:        last modified --
1946:          7/30/91    (map,jpp)
1947:
1948:        purpose --
1949:          writes data to output plot file
1950:       ------------------------------------------------------------------
1951:       dimension xb(3,5), xchg(3)
1952:            sumr2 = 0.0
1953:            do 100 i=1,3
1954:             sumr2 = sumr2 + (xchg(i) - xb(i,1))**2
1955:        100 continue
1956:            r = sqrt(sumr2)
1957:            write(3,*) r,bd,br,bu
1958:            return
1959:            end
1960:       ------------------------------------------------------------------
1961:       call  outpost (dam, bd, br, bu)
1962:
1963:        last modified --
1964:         11/19/92    (dds) - created from out2
1965:
1966:        purpose --
1967:          writes result to output file for bdam postprocessor, bdampost
1968:
1969:        method --
1970:
1971:
1972:        input --
1973:
1974:
1975:        output --
1976:
1977:
1978:        restrictions --
1979:          none set in outpost
1980:       ------------------------------------------------------------------
1981:       call  paus1 (msg)
1982:       business systems integration
1983:       (512)680-3940
1984:       copyright (c) 1988, 1989, 1990, 1991, 1992
```

```
1985:       all rights reserved
1986:       ------------------------------------------------------------------------
1987:       call  postprnt (damaged)
1988:       function:    provides a menu for user selection of postprocess
1989:                    reports.
1990:       parameters:
1991:                    damaged       : logical variable which indicates if
1992:                                    building sustained any damage
1993:                                    t : building sustained damage
1994:                                    f : building sustained no significant
1995:                                        damage (i.e. no component's damage
1996:                                        exceeded 0%)
1997:       ------------------------------------------------------------------------
1998:       call  prephead (itype, report, line_kt)
1999:       function:    writes column headers for the preprocessor reports
2000:                    based on itype and increments the line counter,
2001:                    line_kt, based on number of lines written.
2002:                       itype            type              # report
2003:                         1             wall definitions     2
2004:                         2             component geometry   2
2005:                         3             frame geometry       1
2006:                         4             dependency           1
2007:       parameters:
2008:                    itype    : type of header to be printed
2009:                    report   : indicates which report to print for the
2010:                               specified itype (1 or 2)
2011:                    line_kt : current number of lines written to report file
2012:       ------------------------------------------------------------------------
2013:       call  prepprnt (stat_lun)
2014:       function:    displays main print report menu to allow user to select
2015:                    preprocessor reports to print.  the following reports
2016:                    are generated:
2017:                      component properties
2018:                      wall definitions
2019:                      component geomtry
2020:                      dependencies
2021:       parameters:
2022:                    stat_lun    : logical unit connected to file which will
2023:                                  contain the filenames of all reports files
2024:                                  generated during a preprocessor session
2025:       ------------------------------------------------------------------------
2026:       call  probrep ()
2027:       function:    prints report containing proble description and
2028:                    charge information
2029:       parameters:  none
2030:       ------------------------------------------------------------------------
2031:       call  ptdepend (total_depend)
2032:       function:    saves the id numbers of the independent components
2033:                    into the compg array, col. 14-17
2034:       parameters:
2035:                    total_depend - total number of components in current
2036:                       wall
2037:       ------------------------------------------------------------------------
2038:       call  ptgroup (wall_name, wall_index, itype, id,                -
2039:             set_gen_flag, w_comp_tmp)
2040:       function:    if itype = 1, places the w_comp_table spreadsheet data
2041:                    for master and unique components for the specified
2042:                    wall/roof area into the compg storage area.  if
2043:                    itype = 2, generated components from gencomp are placed
2044:                    in compg.  if itype = 3, frame components are placed from
2045:                    frame_table in compg.  compg contains all required
2046:                    information for each component used in the building.
2047:                    the rows within compg are never physically moved but
2048:                    order is maintained thru the link list mlink.  pointers
```

```
2049:                     to the beginning and end of the master/unique
2050:                     components are maintained in head and tail.  pointers to
2051:                     the beginning and end of the generated nodes are
2052:                     maintained in ghead and gtail.  pointers to the beginning
2053:                     and end of the frame components are maintained in fhead
2054:                     and ftail.
2055:
2056:                     compg contains the following for itypes 1 & 2:
2057:                     column 1 : wall name
2058:                     column 2 : component material type (wood, concrete, etc)
2059:                     column 3 : component type (r/c beam, etc)
2060:                     column 4 : component property name (user assigned)
2061:                     column 5 : component id number
2062:                     column 6 : x1 coordinates of end or corner
2063:                     column 7 : y1 coordinates of end or corner
2064:                     column 8 : x2 coordinates of end or corner
2065:                     column 9 : y2 coordinates of end or corner
2066:                     column 10: master component indicator
2067:                                yes - master
2068:                                no  - unique
2069:                     column 11: center to center spacing
2070:                     column 12: local direction
2071:                                local x - generate in positive x direction
2072:                                local y - generate in positive y direction
2073:                     column 13: number of additional repeated components
2074:
2075:                     compg contains the following for itype 3:
2076:                     column 1 : wall name - always frame
2077:                     column 2 : component material type (concrete or steel)
2078:                     column 3 : component type (r/c frame or steel frame)
2079:                     column 4 : component property name (user assigned)
2080:                     column 5 : component id number
2081:                     column 6 : x1 coordinate of blastward exterior column,
2082:                                coordinates are in the local coordinates of
2083:                                the blastward wall
2084:                     column 7 : y1 coordinate of blastward exterior column,
2085:                                coordinates are in the local coordinates of
2086:                                the blastward wall
2087:                     column 8 : blank
2088:                     column 9 : blank
2089:                     column 10: set to no since no components may be generated
2090:                                from this component
2091:                     column 11: name of blastward wall
2092:                     column 12: total frame height
2093:       parameters:
2094:                     wall_name  - user assigned name of current wall/roof
2095:                     area
2096:                     wall_index - index to wall geometry array
2097:                     itype      - indicates if storing master/unique,
2098:                                  generated or frame components
2099:                                  1 : master/unique component
2100:                                  2 : generated component
2101:                                  3 : frame components
2102:                     id         - id number of master component generated
2103:                                  valid only if itype = 2
2104:                     set_gen_flag - indicates if gen_flag is to be set
2105:                                  valid for generated components,
2106:                                  itype = 2
2107:                                  true - set gen_flag for master component
2108:                                  used for generation to true
2109:                                  false - leave gen_flag alone
2110:                     w_comp_tmp   - temporary storage for the row containing
2111:                        the master component that components are to be
2112:                        generated from.  this is done since the work space,
```

```
2113:                        work_table, is actually the same for master/unique
2114:                        components and for the generated components.
2115:
2116:       ------------------------------------------------------------------------
2117:       call  ptmat (imat, icomp)
2118:       function:    places the comp_table spreadsheet data for the
2119:                    material property group specified by imat and
2120:                    icomp from the xmp storage array.  xmp contains
2121:                    contains all required information for component
2122:                    material property used in the building.  the rows
2123:                    within xmp are never physically moved but order is
2124:                    maintained thru the link list clink.
2125:
2126:                    xmp contains the following:
2127:                    column 1 : icn - identification based on material
2128:                                     and selected subset component
2129:                    column 2 : component property name (user assigned)
2130:                    column 3 : weighting factor
2131:                    column 4 : beam width
2132:                    column 5 : beam thickness
2133:                    column 6 : loaded width
2134:                    column 7 : total weight
2135:                    column 8 : compressive concrete strength
2136:                    column 9 : steel yield strength
2137:                    column 10: depth to tensile steel
2138:                    column 11: area tensile steel
2139:                    column 12: moment of inertia
2140:                    column 13: boundary condition
2141:
2142:       parameters:
2143:                    imat       - material indicator
2144:                                 1 : concrete
2145:                                 2 : steel
2146:                                 3 : masonry
2147:                                 4 : wood
2148:                    icomp      - number of subcategory which occurs under
2149:                                 each material
2150:       ------------------------------------------------------------------------
2151:       call  rclwi (nc1,nel)
2152:
2153:        one way reinforced concrete slabs
2154:
2155:        last modified --
2156:         7/03/91 (jpp)
2157:        01/27/93 (dds) - made moment of inertia be input, xim, rather than
2158:                         calculated value, im and added debugging prints
2159:
2160:        purpose --
2161:         data is read from and/or written to a direct access file.
2162:
2163:        method --
2164:         see theory manual
2165:
2166:        input --
2167:         tape15:  direct access file with individual element data.
2168:
2169:        output --
2170:         tape15 and arguments (see input discussion)
2171:
2172:        restrictions
2173:         none set in rclwi
2174:       ------------------------------------------------------------------------
2175:       call  rc2wi(nc2,nel)
2176:
```

```
2177:        two way reinforced concrete slabs
2178:
2179:        last modified --
2180:         7/03/91 (jpp)
2181:        01/27/93 (dds) - made moment of inertia be input, xim, rather than
2182:                         calculated value, im; added iarch; and added debugging
2183:                         prints
2184:
2185:        purpose --
2186:         data is read from and/or written to a direct access file.
2187:
2188:        method --
2189:         see theory manual
2190:
2191:        input --
2192:         tape15:  direct access file with individual element data.
2193:
2194:        output --
2195:         tape15 and arguments (see input discussion)
2196:
2197:        restrictions
2198:         none set in rc2wi
2199:       ------------------------------------------------------------------------
2200:       call  rcbmi (nc1,nel)
2201:
2202:        reinforced concrete beams
2203:
2204:        last modified --
2205:         7/05/91 (jpp)
2206:        11/18/92 (dds) - made moment of inertia be input, xim, rather than
2207:                         calculated value, im
2208:        01/27/93 (dds) - added debugging prints
2209:
2210:        purpose --
2211:         data is read from and/or written to a direct access file.
2212:
2213:        method --
2214:         see theory manual
2215:
2216:        input --
2217:         tape15:  direct access file with individual element data.
2218:
2219:        output --
2220:         tape15 and arguments (see input discussion)
2221:
2222:        restrictions
2223:         none set in rcbmi
2224:       ------------------------------------------------------------------------
2225:       call  rceci (nc1,nel)
2226:
2227:        exterior reinforced concrete columns
2228:
2229:        last modified --
2230:         7/05/91 (jpp)
2231:        01/27/93 (dds) - made moment of inertia be input, xim, rather than
2232:                         calculated value, im and added debugging prints
2233:
2234:        purpose --
2235:         data is read from and/or written to a direct access file.
2236:
2237:        method --
2238:        see theory manual
2239:
2240:        input --
```

```
2241:          tape15:  direct access file with individual element data.
2242:
2243:         output --
2244:          tape15 and arguments (see input discussion)
2245:
2246:         restrictions
2247:          none set in rceci
2248:        ------------------------------------------------------------------
2249:        call  rcici (ncl,nel)
2250:
2251:         reinforced concrete interior columns
2252:
2253:         last modified --
2254:          7/05/91 (jpp)
2255:         01/27/93 (dds) - made moment of inertia be input, xim, rather than
2256:                          calculated value, im and added debugging prints
2257:
2258:         purpose --
2259:          data is read from and/or written to a direct access file.
2260:
2261:         method --
2262:          in blast vulnerability guide, see section 5.0, figure 5.1*
2263:          and section 6.1                                          *
2264:
2265:         input --
2266:          tape15:  direct access file with individual element data.
2267:
2268:         output --
2269:          tape15 and arguments (see input discussion)
2270:
2271:         restrictions
2272:          none set in rcici
2273:        ------------------------------------------------------------------
2274:        call  rcmrfi (nr,nel)
2275:
2276:         reinforced concrete frames
2277:
2278:         last modified --
2279:          7/12/91 (jpp)
2280:         01/27/93 (dds) - made moment of inertia, xim and height of single story,
2281:                          h, be inputs, rather than calculated values, im and h;
2282:                          added debugging prints,added read for frame col. coord
2283:
2284:         purpose --
2285:          data is read from and/or written to a direct access file.
2286:
2287:         method --
2288:          see theory manual
2289:
2290:         input --
2291:          tape15:  direct access file with individual element data.
2292:           ireclast     last record number in direct access file
2293:
2294:         output --
2295:          tape15 and arguments (see input discussion)
2296:
2297:         restrictions
2298:          none set in rcmrfi
2299:        ------------------------------------------------------------------
2300:        call  rcpsi (ncl,nel)
2301:
2302:         prestressed concrete beams
2303:
2304:         last modified --
```

```
2305:         11/18/92 (dds) - created from rcbmi
2306:
2307:         purpose --
2308:          data is read from and/or written to a direct access file.
2309:
2310:         method --
2311:          in blast vulnerability guide, see section 5.0, figure 5.1*
2312:          and section 6.1                                          *
2313:
2314:         input --
2315:          tape15:  direct access file with individual element data.
2316:
2317:         output --
2318:          tape15 and arguments (see input discussion)
2319:
2320:         restrictions
2321:          none set in rcbmi
2322:        ------------------------------------------------------------------
2323:        call  readat(name,w,xchg,ninc,dxchg,xb,nbop,
2324:            1                  ndep,idep,nr,ireclast,iwrit)
2325:
2326:         last modified --
2327:           7/15/91    (jpp)
2328:          11/19/92    (dds) - added xim to concrete beams read statement
2329:          12/01/92    (dds) - commented out write to screen
2330:          01/27/93    (dds) - modified all required reads and write for all
2331:                              other component types
2332:
2333:         purpose --
2334:           reads input data and calls the proper subroutine
2335:
2336:         method --
2337:
2338:
2339:         input --
2340:
2341:
2342:         output --
2343:
2344:
2345:         restrictions --
2346:           none set in readat
2347:        ------------------------------------------------------------------
2348:        call  readf (formna, usrdat, noinp, ier, ipos, term_key, -
2349:              legalt, numtrm)
2350:        business systems integration
2351:        (512)680-3940
2352:        copyright (c) 1988, 1989, 1990, 1991, 1992
2353:        all rights reserved
2354:        ------------------------------------------------------------------
2355:        call  reflect( zlog, pres, imp )
2356:
2357:         last modified --
2358:           11/16/88   (tkb)   initial release
2359:
2360:         purpose --
2361:           calculates reflected pressure and impulse
2362:
2363:         method --
2364:           uses fitted functions from arbrl-tr-02555 to calculate
2365:           reflected pressure and impulse
2366:
2367:         input --
2368:           variable zlog in argument list
```

```
2369:
2370:        output --
2371:          no files, just variables pres and imp in argument list
2372:
2373:        restrictions --
2374:
2375:          range of applicability:
2376:
2377:             0.170 < z < 100.0    for    z = 10.**zlog
2378:       ------------------------------------------------------------------------
2379:       call  reset ()
2380:       function:     resets required variables, forms, spreadsheets and
2381:                     common blocks to 0s or blanks for beginning of new
2382:                     problem.
2383:       parameters:
2384:                     none
2385:       ------------------------------------------------------------------------
2386:       call  rightj (chosen_value, char_variable, variable_length)
2387:       function:     right justifies data within a variable.  char_variable
2388:                     contains the data to be right justified.  chosen_value
2389:                     contains the right justified data.
2390:       ------------------------------------------------------------------------
2391:       call  rsort2d (array, nval, isortcol, numrow, numcol, idir, -
2392:             ier)
2393:       business systems integration
2394:       (512)680-3940
2395:       copyright (c) 1988, 1989, 1990, 1991, 1992
2396:       all rights reserved
2397:
2398:       function:     sort a real array
2399:       parameters:   array      real array to sort
2400:                     nval       number of values to sort
2401:                     isortcol   column in array sort is based on
2402:                     numrow     number of rows to sort
2403:                     numcol     number of columns in array
2404:                     idir       sort direction (1 ascending, -1 descending)
2405:                     ier        0 : no errors occurred during sort
2406:                                1 : isortcol exceeded number of columns in
2407:                                    array
2408:       ------------------------------------------------------------------------
2409:       log = samecord (x, y, z)
2410:       function:     check if the 2 sets of coordinates specified in x, y,
2411:                     and z are the same.  the function returns a true if
2412:                     if the 2 nodes have the same coordinates and a false
2413:                     if the nodes are different
2414:       parameters:
2415:                     x         - array containing x coordinates for the 2
2416:                                 nodes
2417:                     y         - array containing y coordinates for the 2
2418:                                 nodes
2419:                     z         - array containing z coordinates for the 2
2420:                                 nodes
2421:       ------------------------------------------------------------------------
2422:       call  savedata (input_file_name)
2423:       function:     saves data for current problem which is required
2424:                     for a later restart of the preprocessor.
2425:       parameters:
2426:                     input_file_name     - name of file to save data to;
2427:                                           contains full path
2428:       ------------------------------------------------------------------------
2429:       call  setcid (wall_index, i, j, table, master)
2430:       function:     assigns a unique component id for the specified
2431:                     component.  the component id is generated in the
2432:                     following manner.  the ten thousand and thousand
```

```
2433:                    position are reserved for the wall number, the
2434:                    100s, 10s and 1s position are reserved for the
2435:                    master or unique number a decimal point follows
2436:                    and the decimal places refer to generated number
2437:
2438:       parameters:
2439:                    wall_index - index to wall geometry array
2440:                    i          - number of the master or unique component
2441:                    j          - number of the generated component
2442:                    table      - output variable which will hold the
2443:                                 generated id number
2444:                    master     - true : indicates a master or unique is
2445:                                   being generated so the decimal places
2446:                                   will be 0
2447:                                 false: slave is being generated, number
2448:                                   will be assigned in the decimal places
2449:       ------------------------------------------------------------------
2450:       call  setfid (table)
2451:       function:    assigns a unique frame id for the specified
2452:                    frame.  the frame id is generated in the
2453:                    following manner.  the ten thousand and thousand
2454:                    position are 00 as opposed to regular components where
2455:                    these positions are reserved for the wall number, the
2456:                    100s, 10s and 1s position are reserved for the
2457:                    frame number and the decimal places are 00.
2458:
2459:       parameters:
2460:                    table      - output variable which will hold the
2461:                                 generated id number
2462:       ------------------------------------------------------------------
2463:       call  sideon( zlog, pres, imp )
2464:
2465:        last modified --
2466:          11/16/88   (tkb)   initial release
2467:
2468:        purpose --
2469:          calculates incident pressure and impulse
2470:
2471:        method --
2472:          uses fitted functions from arbrl-tr-02555 to calculate
2473:          pressure and impulse
2474:
2475:        input --
2476:          variable zlog in argument list
2477:
2478:        output --
2479:          no files, just variables pres and imp in argument list
2480:
2481:        restrictions --
2482:
2483:          range of applicability:
2484:
2485:                  0.170 < z < 100.0  for  z = 10.**(zlog)
2486:       ------------------------------------------------------------------
2487:       call  spreadop (chosen_value, option_key, col_position,      -
2488:              row_position, col_width, drow, dcol,                  -
2489:              spread_no, edited_field, term_key, inbuf,             -
2490:              max_rows)
2491:       function:    provides the pop-up options on the spread sheet type
2492:                    forms
2493:       parameters:
2494:                    chosen_value - returns values selected by user or
2495:                                   calculated by program
2496:                    option_key   - option key pressed on spreadsheet
```

```
                    col_position - current column on spreadsheet
                    row_position - current row on spreadsheet
                    col_width    - width of current column
                    drow         - current row position in screen coordinates
                    dcol         - current col. position in screen coordinates
                    spread_no    - number of current spreadsheet
                    edited_field - indicated if current field was edited
                                   0 - field not edited
                                   1 - field edited
                    term_key     - last key user pressed during option
                                   selection
                    inbuf        - character array containing spreadsheet
                                   values
                    max_rows     - maximum row in array used to store
                                   spreadsheet data, not necessarily the
                                   number of rows displayed on the
                                   spreadsheet.  there could be less rows
                                   displayed
      ------------------------------------------------------------------------
      call  spreadop (chosen_value, option_key, col_position,      -
             row_position, col_width, drow, dcol,          -
             spread_no, edited_field, term_key, inbuf,     -
             max_rows)
      function:    provides the pop-up options on the spread sheet type
                   forms
      ------------------------------------------------------------------------
      call  stbmi (ncl,nel)

       steel beams

       last modified --
        7/09/91 (jpp)
       01/27/93 (dds) - added iten variable to read and write and added debuggin
                        prints

       purpose --
        data is read from and/or written to a direct access file.

       method --
        see theory manual

       input --
        tape15:  direct access file with individual element data.

       output --
        tape15 and arguments (see input discussion)

       restrictions
        none set in stbmi
      ------------------------------------------------------------------------
      call  stcdi (ncl,nel)

       corrugated steel decking

       last modified --
        7/09/91 (jpp)
       01/27/93 (dds) - removed b from input variable and made a constant;
                        added debugging prints

       purpose --
        data is read from and/or written to a direct access file.

       method --
        see theory manual
```

```
2561:
2562:        input --
2563:         tape15:  direct access file with individual element data.
2564:
2565:        output --
2566:         tape15 and arguments (see input discussion)
2567:
2568:        restrictions
2569:         none set in stcdi
2570:       ------------------------------------------------------------------
2571:       call  steci (nc1,nel)
2572:
2573:        steel exterior columns
2574:
2575:        last modified --
2576:         7/10/91 (jpp)
2577:        01/27/93 (dds) - added iten variable to read and write and added debuggin
2578:                         prints
2579:
2580:        purpose --
2581:         data is read from and/or written to a direct access file.
2582:
2583:        method --
2584:         see theory manual
2585:
2586:        input --
2587:         tape15:  direct access file with individual element data.
2588:
2589:        output --
2590:         tape15 and arguments (see input discussion)
2591:
2592:        restrictions
2593:         none set in steci
2594:       ------------------------------------------------------------------
2595:       call  stici (nc1,nel)
2596:
2597:        steel interior columns
2598:
2599:        last modified --
2600:         7/11/91 (jpp)
2601:        01/27/93 (dds) - added debugging debugging prints
2602:
2603:        purpose --
2604:         data is read from and/or written to a direct access file.
2605:
2606:        method --
2607:         in blast vulnerability guide, see section 5.0, figure 5.1*
2608:         and section 6.1                                          *
2609:
2610:        input --
2611:         tape15:  direct access file with individual element data.
2612:
2613:        output --
2614:         tape15 and arguments (see input discussion)
2615:
2616:        restrictions
2617:         none set in stici
2618:       ------------------------------------------------------------------
2619:       call  stmrfi (nr,nel)
2620:
2621:        steel frames
2622:
2623:        last modified --
2624:         7/12/91 (jpp)
```

```
2625:        01/27/93 (dds) - made single story height, h, be input; removed idc
2626:                         as input; removed calc of h; added debugging prints
2627:                         added read for col.coord.
2628:
2629:        purpose --
2630:         data is read from and/or written to a direct access file.
2631:
2632:        method --
2633:        see theory manual
2634:
2635:        input --
2636:         tape15:  direct access file with individual element data.
2637:
2638:          ireclast     last record number in direct access file
2639:
2640:        output --
2641:         tape15 and arguments (see input discussion)
2642:
2643:        restrictions
2644:         none set in stmrfi
2645:       ------------------------------------------------------------------------
2646:       call  stmswi (ncl,nel)
2647:
2648:        metal stud walls
2649:
2650:        last modified --
2651:         7/12/91 (jpp)
2652:        01/27/93 (dds) - added debugging prints, all reference to ioptbc=2
2653:                         removed
2654:
2655:        purpose --
2656:         data is read from and/or written to a direct access file.
2657:
2658:        method --
2659:         see theory manual
2660:
2661:        input --
2662:         tape15:  direct access file with individual element data.
2663:
2664:        output --
2665:         tape15 and arguments (see input discussion)
2666:
2667:        restrictions
2668:         none set in stmswi
2669:       ------------------------------------------------------------------------
2670:       call  stowji (ncl,nel)
2671:
2672:        open web steel joists
2673:
2674:        last modified --
2675:         7/08/91 (jpp)
2676:        01/27/93 (dds) - removed variables from read and write which are no
2677:                         longer required; added debugging prints, all
2678:                         reference to web shear failure deleted
2679:
2680:        purpose --
2681:         data is read from and/or written to a direct access file.
2682:
2683:        method --
2684:         see theory manual
2685:
2686:        input --
2687:         tape15:  direct access file with individual element data.
2688:
```

```
2689:         output --
2690:          tape15 and arguments (see input discussion)
2691:
2692:         restrictions
2693:          none set in stowji
2694:        ------------------------------------------------------------------
2695:        call  sum(dam,bd,br,bu)
2696:
2697:         last modified --
2698:           7/10/91    (map,jpp)
2699:
2700:         purpose --
2701:           summation of building damage
2702:
2703:         method --
2704:
2705:
2706:         input --
2707:
2708:
2709:         output --
2710:
2711:
2712:         restrictions --
2713:           none set in sum
2714:        ------------------------------------------------------------------
2715:        call  threat (r, w, iopt2)
2716:
2717:          last modified --
2718:            8/16/90    (maq)
2719:
2720:          purpose --
2721:            definition of charge weight and standoff from common building
2722:
2723:          method --
2724:            charge weight and standoff are directly input
2725:
2726:          input --
2727:            none
2728:
2729:          output --
2730:            charge weight and standoff
2731:
2732:          restrictions
2733:            charge weight must be a value between 35.0 and 4000.0 pounds
2734:        ------------------------------------------------------------------
2735:        call  updatgen (wall_name, wall_index, ier)
2736:        function:     checks all master/unique components for the specified
2737:                      wall to determine if any master components have not
2738:                      been generated.  if a master is found whose components
2739:                      have not been generated, the components will be generated
2740:                      and the generated flag updated.
2741:        parameters:
2742:                      wall_name  - name of current wall/roof area
2743:                      wall_index - index into the wall area used to
2744:                                   access wall number for current wall_name
2745:                      ier        - 0 : no error occurred
2746:                                   1 : one or more required pieces of data
2747:                                       in missing from the current master
2748:        ------------------------------------------------------------------
2749:        call  validchg (lun, fatal_error, error_occurred,           -
2750:              total_fatal, total_warnings)
2751:        function:     check the position of the charge to determine if the
2752:                      charge's location meets minimum requirements of program
```

these are that the scaled standoff is greater than 1.0 ft/lb^(1/3) and that charge cannot be inside building or over roof. also issues warnings when charge is within scaled standoff of 3.0 ft/lb^(1/3) or when charge height off ground to standoff ratio is less than 5.0 since this may invalidate surface burst assumption in blast loads calculations.

parameters:

lun	: logical unit number of file messages are written to if to_screen if false
fatal_error	: logical variable indicating if fatal error occurred. this variable is initialized once at the beginning of the validation process to false and set to true each time a fatal error occurs f - no error occurred t - fatal error occurred
error_occurred	: logical variable indicating if a fatal error occurred in the checking done by this subroutine
total_fatal	: total number of fatal errors
total_warnings	: total number of warning errors

---

call validcmp (lun, fatal_error, error_occurred, - total_fatal)

function: performs validation checking of component properties.
1. no component has an undefined value.
2. no component has a value less than or equal to 0.
3. components have unique name for each category.

parameters:

lun	- logical unit number for error output file
fatal_error	- logical variable indicating if fatal error occurred. this variable is initialized once at the beginning of the validation process to false and set to true each time a fatal error occurs f - no error occurred t - fatal error occurred
error_occurred	- indicates error occurred in this category
total_fatal	- total number of fatal errors

---

call validpnd (lun, fatal_error, error_occurred, - total_fatal, total_warnings)

function: performs validation checking of dependency data. the following checks are made:
1. total number of dependency pairs are checked to see if they exceed the maximum allowable number of pairs (fatal)
2. each wall/roof area is checked to see if the dependencies for that area have been generated. (warning)
3. each row of compg is checked to see make certain that all independent ids in the row are unique within that row. (fatal)
4. each independent id is checked to make sure that it has a corresponding dependent id within compg. (fatal)

parameters:

lun	: logical unit number of file messages are written to if to_screen if false
fatal_error	: logical variable indicating if fatal error occurred. this variable is

```
                                        initialized once at the beginning of
                                        the validation process to false and set
                                        to true each time a fatal error occurs
                                          f - no error occurred
                                          t - fatal error occurred
                    error_occurred : logical variable indicating if a
                                        fatal error occurred in the checking
                                        done by this subroutine
                    total_fatal    : total number of fatal errors
                    total_warnings : total number of warning errors
      ------------------------------------------------------------------------
      call  validwal (lun, fatal_error, error_occurred,           -
            total_fatal, total_warnings)
      function:     performs validation checking of wall data.
                    1.  no walls may have identical coordinates
                    2.  wall names must be unique and no wall/roof may
                        be named frame.
                    3.  no blank coordinates
                    4.  coordinates in a wall must be planar
                    5.  coordinates were entered in clockwise or
                        counterclockwise order
                    6.  y coordinates must be non-negative
                    7.  3rd coordinate must > 4th coordinate
                    8.  walls must connect
      parameters:
                    lun            : logical unit number for error output
                                        file
                    fatal_error    : logical variable indicating if fatal
                                        error occurred.  this variable is
                                        initialized once at the beginning of
                                        the validation process to false and set
                                        to true each time a fatal error occurs
                                          f - no error occurred
                                          t - fatal error occurred
                    error_occurred : indicates error occurred in this
                                        category
                    total_fatal    : total number of fatal errors
                    total_warnings : total number of warning errors
      ------------------------------------------------------------------------
      call  validwcmp (lun, fatal_error, error_occurred,          -
            total_fatal, total_warnings)
      function:     performs validation checking of wall component
                    definition data.
                    1.  check if component type is defined
                    2.  check if component property name is defined
                    3.  check if mismatch occurs between material type
                        and component type
                    4.  check if mismatch occurs between component type
                        and component property name
                    5.  check if component property name for each item in
                        compg is in xmp table
                    6.  check for component specific type errors (done
                        by itypechk which is called by chkcomp)
                    6.  coordinates for a given wall must be unique
                    7.  coordinates of component must be in bounds of
                        wall.
                    8.  walls must connect
                    9.  coordinates must define legitimate beam or plate
      parameters:
                    lun            : logical unit number of file messages
                                        are written to if to_screen if false
                    fatal_error    : logical variable indicating if fatal
                                        error occurred.  this variable is
                                        initialized once at the beginning of
```

```
                                        the validation process to false and set
                                        to true each time a fatal error occurs
                                          f - no error occurred
                                          t - fatal error occurred
                    error_occurred : logical variable indicating if a
                                        error occurred in the checking
                                        done by this subroutine
                    total_fatal    : total number of fatal errors
                    total_warnings : total number of warning errors
      ------------------------------------------------------------------------
      int = wallchk (wall_index)
      function:     to check if four nodes in wall area in row "wall_index"
                    are coplanar and if four nodes are entered either
                    clockwise or counterclockwise and print error message
                    detailing any error that is located
                    if there are no problems the wallchk = 0
                    if there are problems, wallchk = 1
      parameters:
                    wall_index   - index into the wall area used to
                       access wall number for current wall_name
      ------------------------------------------------------------------------
      call  walldef (ier)
      function:     allows user input of the wall/roof areas.  user
                    inputs a 10 character unique name for the area
                    followed by the global x, y and z coorindates for
                    the four corner points.  a maximum of 50 unique wall/roof
                    areas are allowed.  this maximum is set in sprdhead.dat.
                    the dimensioning of wall_table allows for up to
                    max_row_ever columns.  the following checks are
                    performed by walldef:
                      1.  checks if user changed name of a wall/roof area
                          so that compg can be updated.  the old name
                          becomes the new name in compg.
                      2.  checks for a deleted row in the wall/roof table
                          so that compg is kept current.  all references
                          to that wall/roof area are deleted from compg.
                      3.  checks that all wall/roof areas have unique names.
                      4.  checks that no wall/roof area has the name frame,
                          since this is a reserved word.
                      5.  checks if 2 wall/roof areas have identical
                          coordinates.
                      6.  checks if a wall/roof area has 2 identical nodes.
                      7.  checks for overlapping or gaps in wall/roof areas

                    wall_table contains the following data in each column:

                     1.  user defined unique name (user entry)
                     2.  global x coordinate for 1st corner point (user entry)
                     3.  global y coordinate for 1st corner point (user entry)
                     4.  global z coordinate for 1st corner point (user entry)
                     5.  global x coordinate for 2nd corner point (user entry)
                     6.  global y coordinate for 2nd corner point (user entry)
                     7.  global z coordinate for 2nd corner point (user entry)
                     8.  global x coordinate for 3rd corner point (user entry)
                     9.  global y coordinate for 3rd corner point (user entry)
                     10. global z coordinate for 3rd corner point (user entry)
                     11. global x coordinate for 4th corner point (user entry)
                     12. global y coordinate for 4th corner point (user entry)
                     13. global z coordinate for 4th corner point (user entry)
                     14. wall id number (program use)
                     15. characters 1-3 number of components for wall
                           area (program use)
                         characters 8-9 used for roof connectivity in
                           bldgchk (program use)
```

```
                        character 10   used to indicate if dependencies have
                          been generated:
                            t - dependencies have been generated for
                                current wall/roof area
                            f - dependencies have not been generated for
                                current wall/roof area
      parameters:
                   ier          - 0 : no errors occurred
                                  1 : duplicate wall/roof names
                                  2 : 2 wall/roof areas have identical
                                      coordinates
                                  3 : a wall/roof area has 2 identical
                                      nodes
                                  4 : wall/roof area not coplanar
                                  5 : overlap or gap in wall/roof area(s)
                                  6 : problem in connectivity
      ------------------------------------------------------------------------
      log =n wallok (lun, to_screen, fatal_error,              -
             error_occurred, total_fatal, total_warnings)
      function:    checks if the wall/roof areas match together at corners
                   to form a building without gaps or overlapping wall
                   areas. if a problem is found, the function returns
                   false after printing out an error message.  a true
                   issued by this subroutine does not guarentee that the
                   building is well defined, but a false is indicative
                   that there is a problem.
      parameters:
                   lun             : logical unit number of file messages
                                     are written to if to_screen if false
                   to_screen       : logical variable indicates where
                                     error messages will be displayed
                                     t : display to screen
                                     f : send to file indicated by logical
                                         unit lun
                   fatal_error     : logical variable indicating if fatal
                                     error occurred.  this variable is
                                     initialized once at the beginning of
                                     the validation process to false and set
                                     to true each time a fatal error occurs
                                       f - no error occurred
                                       t - fatal error occurred
                   error_occurred : logical variable indicating if a
                                     fatal error occurred in the checking
                                     done by this subroutine
                   total_fatal     : total number of fatal errors
                   total_warnings : total number of warning errors
      ------------------------------------------------------------------------
      call  wallrep ()
      function:    generates the wall definition's report sorted by the
                   wall name.  two reports are required to provide all
                   information.  the first report prints the wall/roof
                   name with coordinates of corner 1 and corner 2.  the
                   second report prints the wall/roof name with
                   coordinates of corner 3 and corner 4.
      parameters:
                   none
      ------------------------------------------------------------------------
      call  wdamrep (wall_name)
      function:    prints report on damaged components for specified wall.
                   one of two reports is generated, depending on if the
                   it is a standard wall/roof area or a frame.  the report
                   contains the following information:
                     for non-frame components:
                        damaged building components report -
```

```
                              component type
                              percent damage
                              component local coordinates
                              repair or replace
                              p-i diagram terms, pbar
                              p-i diagram terms, ibar
                         for frame components:
                            damaged building components report -
                              component type
                              percent damage
                              blastward wall name
                              component local coordinates
                              repair or replace
                              p-i diagram terms, pbar
                              p-i diagram terms, ibar
      parameters:
                   wall_name       : user assigned name of wall/roof area,
                                     unless the area is a frame in which
                                     case the program assigned the area name
                                     of frame
      ------------------------------------------------------------------------
      call  wdbmi (nc1,nel)

       wood beams

       last modified --
        7/08/91 (jpp)
       01/28/93 (dds) - made moment of inertia be input, xim, rather than
                        calculated value, im; added sigy; added debugging
                        prints;
                        modified ci and cp

       purpose --
        data is read from and/or written to a direct access file.

       method --
        see theory manual

       input --
        tape15:  direct access file with individual element data.

       output --
        tape15 and arguments (see input discussion)

       restrictions
        none set in wdbmi
      ------------------------------------------------------------------------
      call  wdeci (nc1,nel)

       exterior wood columns

       last modified --
        7/08/91 (jpp)
       01/27/93 (dds) - made moment of inertia be input, xim, rather than
                        calculated value, xim; added sigy; added debugging print
                        modified ci and cp

       purpose --
        data is read from and/or written to a direct access file.

       method --
        see theory manual

       input --
```

```
3073:          tape15:  direct access file with individual element data.
3074:
3075:         output --
3076:          tape15 and arguments (see input discussion)
3077:
3078:         restrictions
3079:          none set in wdeci
3080:        ------------------------------------------------------------------
3081:        call  wdici (nc1,nel)
3082:
3083:         interior wood columns
3084:
3085:         last modified --
3086:          7/09/91 (jpp)
3087:         01/28/93 (dds) - made moment of inertia be input, im, rather than
3088:                          calculated value; added debugging prints
3089:
3090:         purpose --
3091:          data is read from and/or written to a direct access file.
3092:
3093:         method --
3094:          see theory manual
3095:
3096:         input --
3097:          tape15:  direct access file with individual element data.
3098:
3099:         output --
3100:          tape15 and arguments (see input discussion)
3101:
3102:         restrictions
3103:          none set in wdici
3104:        ------------------------------------------------------------------
3105:        call  wdrfi (nc1,nel)
3106:
3107:         wood roofs
3108:
3109:         last modified --
3110:          7/11/91 (jpp)
3111:         01/28/93 (dds) - added sigy; added debugging prints
3112:                          modified ci and cp
3113:
3114:         purpose --
3115:          data is read from and/or written to a direct access file.
3116:
3117:         method --
3118:          see theory manual
3119:
3120:         input --
3121:          tape15:  direct access file with individual element data.
3122:
3123:         output --
3124:          tape15 and arguments (see input discussion)
3125:
3126:         restrictions
3127:          none set in wdrfi
3128:        ------------------------------------------------------------------
3129:        call  wdwli (nc1,nel)
3130:
3131:         wood walls
3132:
3133:         last modified --
3134:          7/11/91 (jpp)
3135:         01/28/93 (dds) - added sigy to read and write and added debugging
3136:                          prints
```

```
                        modified ci and cp

      purpose --
       data is read from and/or written to a direct access file.

      method --
       see theory manual

      input --
       tape15:  direct access file with individual element data.

      output --
       tape15 and arguments (see input discussion)

      restrictions
       none set in wdwli
     ------------------------------------------------------------------------
     int =wnindex (comp_id)
     function:    strips the wall area number off the component id number
                  and searches wall_table until it finds the row with
                  this wall area number in column 14.  the function name
                  wnindex returns the row number.  if the id number is
                  for a frame component (00 are first 2 digits), the
                  index into wall_table for the blastward wall is returned
                  in the function name
     parmaeters :
                  comp_id              : component id which needs a
                                         corresponding wall index
     ------------------------------------------------------------------------
     call  wrthead (lun, itype, line_kt)
     function:    writes column headers based on itype and increments the
                  line counter, line_kt, based on number of lines written.
                     itype             subroutine     report
                       1                 mbamrep        1
                       2                 wdamrep        1 non-frame
                       3                 wdamrep        1 frame
                       4                 blstrep        1 non-frame
                       5                 blstrep        1 frame
     parameters:
                  lun     : logical unit connected to report file
                  itype   : type of header to be printed
                  line_kt : current number of lines written to report file
     ------------------------------------------------------------------------
     call  xylimit (to_screen, lun, wall_index, xmax, ymax,      -
            xmin, ymin, ier)

     function:    to calculate the maximum and minimum local coordinates
                  in the wall area in the row of wall_table indentified
                  by the pointer wall_index
     parameters:
                  to_screen       - logical variable indicates where
                                    error messages will be displayed
                                    t : display to screen
                                    f : send to file indicated by logical
                                        unit lun
                  lun             - logical unit number for error output
                                    file
                  wall_index      - index into the wall area used to
                                    access wall number for current
                                    wall_name
                  xmax            - maximum x value for specified wall
                  ymax            - maximum y value for specified wall
                  xmin            - minimum x value for specified wall
                  ymin            - minimum y value for specified wall
```

```
                    ier             - error flag
                                      0 : no errors occurred
                                      1 : all coordinates for current wall
                                          were 0
                                      2 : error occurred converting wall
                                          coordinates from character to
                                          real
                                      3 : wall name not defined
      ------------------------------------------------------------------------
      call  xyz2 (windex, j, xyz, x)
      function:    finds global x,y,z coordinates of j local coordinates
                   in vector x and places them in vector xyz.  the local
                   coordinates in x are in the wall area designated by
                   the row pointer windex in wall_table.  each point in
                   x has 2*j coordinates.  each point in xyz has 3*j
                   coordinates.
      parameters:
                   windex : index into the wall area used to access wall
                            number for current wall_name
                   j      : number of coordinates sent; should be no
                            more than 4 pairs of coordinates
                   xyz    : global coordinates
                   x      : local coordinates
      ------------------------------------------------------------------------
      call  xyzcoord (mindex, xyz, id)
      function:    finds global x,y,z coordinates of two component endpoints
                   or corner points for component in row 'mindex' in
                   compg matrix if mindex > 0
                   if mindex=0, then the the variable 'id', which should
                   contain the the component id number, is used to determine
                   mindex
                   (xyz(i),i=1,3) global coordinates of first point
                   (xyz(i),i=4,6) global coordinates of second point
      parameters:
                   mindex :  flag indicating to use component id or row #
                               0 - flag to use component id to find desired
                                   component in compg and place component
                                   endpoints in xyz
                              >0 - row in compg for coordinates to be
                                   placed in xyz
                   xyz    :  global coordinates of the two endpoints
                             of the desired component
                   id     :  id number of desired component, only used
                             if mindex = 0
```

Appendix C

Description of Primary Subroutines Called in the FACEDAP Code that are Unique to Single Component Analysis

Revision 1.2
5/20/94

```
   1:       ------------------------------------------------------------------------
   2:       call  bloadc ()
   3:       function:    This routine is designed for future expansion when
   4:                    the user is allowed to enter the charge information
   5:                    by charge weight, standoff and free-field or reflected
   6:                    or by blast pressure and impulse
   7:       parameters:  none
   8:       ------------------------------------------------------------------------
   9:       call  comprep ()
  10:       function:    Prints report on component damage for Single Component
  11:                    Analysis
  12:       parameters:  none
  13:       ------------------------------------------------------------------------
  14:       call  cprphead (lun, itype, report, line_kt)
  15:       function:    Writes column headers for the COMPONENT Preprocessor
  16:                    Component Property reports based on itype and report and
  17:                    increments the line counter, line_kt, based on number
  18:                    of lines written.
  19:                       itype            property       # reports
  20:                         1               RCBMI             2
  21:                         2               RC1WI             2
  22:                         3               RC2WI             2
  23:                         4               RCECI             2
  24:                         5               RCICI             2
  25:                         6               RCMRFI            2
  26:                         7               RCPSI             2
  27:                         8               STBMI             2
  28:                         9               STMSWI            1
  29:                        10               STOWJI            1
  30:                        11               STCDI             2
  31:                        12               STECI             2
  32:                        13               STICI             2
  33:                        14               STMRFI            2
  34:                        15               MAU1WI            2
  35:                        16               MAU2WI            2
  36:                        17               MAR1WI            2
  37:                        18               MAR2WI            2
  38:                        19               MAPILI            2
  39:                        20               WDWLI             2
  40:                        21               WDRFI             2
  41:                        22               WDBMI             2
  42:                        23               WDECI             2
  43:                        24               WDICI             2
  44:       parameters:
  45:                    lun      : logical unit connected to report file
  46:                    itype    : type of header to be printed
  47:                    report   : indicates which report to print for the
  48:                               specified itype (1 or 2)
  49:                    line_kt  : current number of lines written to report file
  50:       ------------------------------------------------------------------------
  51:       call  filxist2 (input_file_name, ier)
  52:       routine     : FILXIST2
  53:       function    : Allows user to select the COMPPREP file name to save
  54:                     the file under.  The default is the current file name.
  55:                     Issues warning message if the selected file already
  56:                     exists.  User then has the choice of writing over the
  57:                     existing file or aborting the save.
  58:       parameters :
  59:                     input_file_name     : Current Preprocessor file name
  60:                                           selected by user.  Contains full
  61:                                           path
  62:                     ier                 : 0 - Save file
  63:                                           : 1 - Do not save file
  64:       ------------------------------------------------------------------------
```

```
call  getcomp (input_file_name)
function:    Retrieve data for current problem which is required
             for a restart of the COMPONENT Preprocessor, COMPPREP.
parameters:
             input_file_name     : name of file to retrive data
                                   from; contains full path
-------------------------------------------------------------------------
call  getdefct (mat_name, menu, num_comps)
function:    Provides a pop-up menu for COMPPREP of all component
             types which has properties defined for them.  This is
             done by checking the XMP array.
parameters:  mat_name  - material name of selected material
             menu      - array to store component types that have
                         been used
             num_comps - total number of component types that have
                         been used
-------------------------------------------------------------------------
call  getdefmt (menu, num_mats)
function:    Provides a pop-up menu for COMPPREP of all materials
             which has properties defined for them.  This is done
             by checking the XMP array.
parameters:  menu      - array to store material names that have
                         been used
             num_mats - total number of materials that have been used
-------------------------------------------------------------------------
call  getspred (spread_no, ier)
function:    Reads all required variables for the specified
             spreadsheets from the data file SPRDHED2.DAT.
             This routine is used by COMPPREP.
             The following values are read:
             1. Spread no
             2. Number of columns on spreadsheet
             3. Number of rows on spreadsheet
             4. Number of column header lines
             5. Column Headers
             6. Help text
             7. Variable type for each column (i.e. character,
                integer, real)
             8. Number of defaults
             9. Column numbers of columns which have default settings
            10. Default value for the default columns
            11. Fields where duplication on spreadsheet allowed
                0 - field can be duplicated
                1 - field can not be duplicated
parameters:
             spread_no   - number of spreadsheet to retrieve data for
             ier         - 0 : no error occurred
                           1 : error occurred opening file,
                               SPRDHEAD.DAT
                           2 : error occurred reading SPRDHED2.DAT
-------------------------------------------------------------------------
call  getvars (spread_no, col_position, calc_type)
function:    Used by spreadop for determining the columns used in
             default calculations.  Based on the spreadsheet number,
             spread_no and the column position on the spreadsheet,
             col_position, the required column numbers are returned
             along with an indicator of the equation to be used for
             calcluating.  This routine is used for the Component
             Analysis, COMPPREP.
-------------------------------------------------------------------------
call  gtshort (comp_type_name, icomp, imat, ier)
function:    Uses the specified component property name to find
             the icomp and imat that corresponds to the specified
             type.  Comp_type_name is the full name.
```

```
      parameters:
                    comp_type_name : name of component type
                    icomp          : component # with a given material group
                    imat           : material #
                    ier            : 0 - no error occurred
                                     1 - component type name not found
      ------------------------------------------------------------------------
      call  initcomp ()
      function:     Initializes the COMPDEF.FRM with the first component
                    it finds in XMP.
      parameters:   none
      ------------------------------------------------------------------------
      call  inpfil2 (file_name, pattern, ier)
      function:     This routine was modified from the IOSUB INPFIL routine.
                    This routine has been specifically modified to strip off
                    the .BLG extension.  Calling with any other extension
                    will require modification of this code
      parameters:   file_name  file to search for
                    pattern    input pattern
                    ier        error code, 0=normal, 1=error
      ------------------------------------------------------------------------
      call  optionb (usrdat, key_pressed, form_number, ipos)
      function:     Provides the options for forms in BDAMPREP and COMPPREP.
      parameters:
                    usrdat         : array containing form data
                    key_pressed    : keystroke user left form with and
                                     is used as an input parameter into
                                     this routine to determine action to
                                     take
                    form_number    : number of form being edited
                    ipos           : current row number in form (usrdat)
                                     when entered this routine
      ------------------------------------------------------------------------
      call  probrep2 ()
      function:     Prints report containing problem description and
                    load information for the Single Component Preprocessor
      parameters:   none
      ------------------------------------------------------------------------
      call  resetc ()
      function:     Resets required variables, forms, spreadsheets and
                    common blocks to 0s or blanks for beginning of new
                    problem for the Component Preprocessor, COMPPREP.
      parameters:
                    none
      ------------------------------------------------------------------------
      call  savecomp (input_file_name)
      function:     Saves data for current problem which is required
                    for a later restart of the COMPONENT Preprocessor,
                    COMPPREP.
      parameters:
                    input_file_name    - name of file to save data to;
                                         contains full path
```